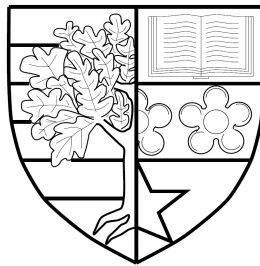


EVOLUTIONARY ALGORITHMS AND WEIGHTING
STRATEGIES FOR FEATURE SELECTION IN
PREDICTIVE DATA MINING

by

Manjula Dissanayake



Submitted for the degree of
Doctor of Philosophy

DEPARTMENT OF COMPUTER SCIENCE
SCHOOL OF MATHEMATICAL AND COMPUTER SCIENCES
HERIOT-WATT UNIVERSITY

May 2015

The copyright in this thesis is owned by the author. Any quotation from the report or use of any of the information contained in it must acknowledge this report as the source of the quotation or information.

Abstract

The improvements in Deoxyribonucleic Acid (DNA) microarray technology mean that thousands of genes can be profiled simultaneously in a quick and efficient manner. DNA microarrays are increasingly being used for prediction and early diagnosis in cancer treatment. Feature selection and classification play a pivotal role in this process. The correct identification of an informative subset of genes may directly lead to putative drug targets. These genes can also be used as an early diagnosis or predictive tool. However, the large number of features (many thousands) present in a typical dataset present a formidable barrier to feature selection efforts.

Many approaches have been presented in literature for feature selection in such datasets. Most of them use classical statistical approaches (e.g. correlation). Classical statistical approaches, although fast, are incapable of detecting non-linear interactions between features of interest. By default, Evolutionary Algorithms (EAs) are capable of taking non-linear interactions into account. Therefore, EAs are very promising for feature selection in such datasets.

It has been shown that dimensionality reduction increases the efficiency of feature selection in large and noisy datasets such as DNA microarray data. The two-phase Evolutionary Algorithm/ k -Nearest Neighbours (EA/ k -NN) algorithm is a promising approach that carries out initial dimensionality reduction as well as feature selection and classification.

This thesis further investigates the two-phase EA/ k -NN algorithm and also introduces an adaptive weights scheme for the k -Nearest Neighbours (k -NN) classifier. It also introduces a novel weighted centroid classification technique and a correlation guided mutation approach. Results show that the weighted centroid approach is capable of out-performing the EA/ k -NN algorithm across five large biomedical datasets. It also identifies promising new areas of research that would complement the techniques introduced and investigated.

Acknowledgements

I am very grateful for the support that I received throughout this research. First of all, I would like to thank Professor David Corne for his continued guidance and encouragement. I always felt supported and confident under the supervision of Professor Corne. I would like to thank my parents for their ever-present support and for encouraging me to always pursue my academic goals. I would also like to thank my wife, Indy, for her patience and understanding throughout what has been a long process. Finally, I would like to thank the rest of my family and friends for their support, and my brother, Upul Dissanayaka, for helping to proofread this work.

Declaration Statement

This is where the declaration form is going to go...

Contents

| | | |
|----------|---|----------|
| 1 | Introduction | 1 |
| 1.1 | Introduction | 1 |
| 1.1.1 | Feature Selection and Classification | 2 |
| 1.2 | Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm for Feature Selection and Classification | 3 |
| 1.3 | Datasets | 4 |
| 1.3.1 | Leukaemia Dataset | 4 |
| 1.3.2 | Ovarian Cancer Dataset | 5 |
| 1.3.3 | Prostate Cancer Dataset | 5 |
| 1.3.4 | Breast Cancer Dataset | 5 |
| 1.3.5 | Colon Cancer Dataset | 5 |
| 1.4 | Contributions | 6 |
| 1.5 | Publications Resulting From this Research | 7 |
| 1.6 | Outline of Thesis | 7 |
| 2 | Literature Review | 9 |
| 2.1 | Deoxyribonucleic Acid Microarrays | 9 |
| 2.2 | Overview of Feature Selection and Classification Techniques | 10 |
| 2.2.1 | Filter Techniques | 11 |
| 2.2.2 | Wrapper Techniques | 11 |
| 2.2.3 | Embedded Techniques | 13 |
| 2.2.4 | Hybrid Techniques | 13 |
| 2.3 | Evolutionary Algorithm/ k -Nearest Neighbours Algorithm | 15 |
| 2.4 | Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm | 16 |
| 2.4.1 | Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Al- gorithm Explained | 18 |

| | | |
|----------|---|-----------|
| 2.4.2 | The Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours | |
| | Experimental Design | 21 |
| 2.4.2.1 | Phase One | 21 |
| 2.4.2.2 | Phase Two | 23 |
| 2.4.3 | Results from Two-Phase Evolutionary Algorithm/ k -Nearest | |
| | Neighbours Algorithm | 24 |
| 2.5 | Multi-Objective Evolutionary Algorithms | 25 |
| 2.6 | Related Results from Literature for the Datasets Used in This Thesis | 28 |
| 2.6.1 | Leukaemia Dataset | 28 |
| 2.6.2 | Prostate Cancer Dataset | 32 |
| 2.6.3 | Ovarian Cancer Dataset | 32 |
| 2.6.4 | Colon Cancer Dataset | 32 |
| 2.6.5 | Breast Cancer Dataset | 32 |
| 3 | Study of Parameters and Multi-Objective Approach in Two-Phase | |
| | Evolutionary Algorithm/k-Nearest Neighbours Algorithm | 34 |
| 3.1 | Algorithms | 35 |
| 3.1.1 | Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours . . | 35 |
| 3.1.2 | Multi-Objective Two-Phase Evolutionary Algorithm/ k -Nearest | |
| | Neighbours | 39 |
| 3.2 | Datasets | 41 |
| 3.3 | Experiments | 42 |
| 3.3.1 | Leukaemia Dataset | 42 |
| 3.3.1.1 | Single Objective Two-Phase Evolutionary Algorithm/ k - | |
| | Nearest Neighbours | 42 |
| 3.3.1.2 | Multi-Objective Two-Phase Evolutionary Algorithm/ k - | |
| | Nearest Neighbours | 44 |
| 3.3.2 | Ovarian Cancer & Prostate Cancer Datasets | 44 |
| 3.4 | Results & Discussion | 44 |
| 3.4.1 | Initial Parameter Tuning: Chromosome Size | 44 |
| 3.4.2 | Initial Parameter Tuning: The Effect of α | 46 |
| 3.4.2.1 | Effect of α on the Leukaemia Dataset | 51 |
| 3.4.2.2 | Effect of α on the Prostate Cancer Dataset | 51 |
| 3.4.3 | Leukaemia Dataset | 51 |

| | | |
|----------|---|-----------|
| 3.4.4 | Ovarian Cancer Dataset | 54 |
| 3.4.5 | Prostate Cancer Dataset | 55 |
| 3.5 | Conclusion | 56 |
| 4 | An Adaptive Weights Scheme for k-Nearest Neighbours in Evolutionary Algorithm/k-Nearest Neighbours Algorithm for Feature Selection | 58 |
| 4.1 | Introduction | 58 |
| 4.2 | Improvements to the Methodology | 59 |
| 4.2.1 | Cross-validation | 59 |
| 4.2.2 | Model Selection | 60 |
| 4.2.3 | Setting up of Phase I & II of the Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm | 62 |
| 4.3 | Adaptive Weights for k -Nearest Neighbours | 63 |
| 4.3.1 | Adaptive Weights for k -Nearest Neighbours Explained | 64 |
| 4.3.2 | Application of Adaptive Weights | 65 |
| 4.4 | Algorithms | 67 |
| 4.4.1 | Stand-alone k -Nearest Neighbours and Weighted k -Nearest Neighbour | 67 |
| 4.4.2 | Single-objective Evolutionary Algorithm/ k -Nearest Neighbours Algorithm | 68 |
| 4.4.3 | Single-objective Evolutionary Algorithm/Weighted- k -Nearest Neighbours Algorithm | 68 |
| 4.4.4 | Single-objective Evolutionary Algorithm/Weighted Centroid Classification Algorithm | 69 |
| 4.5 | Preliminary Experiments | 70 |
| 4.5.1 | Speed of the Algorithm | 70 |
| 4.5.2 | Estimating T | 72 |
| 4.6 | Main Experiments | 73 |
| 4.6.1 | Set-up of the Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm | 73 |
| 4.6.2 | Exploration of Feature Weights | 74 |
| 4.7 | Results and Discussion | 74 |
| 4.7.1 | Speed of the Algorithm | 74 |

| | | |
|---------|--|----|
| 4.7.2 | Estimating T: Stand-alone k -Nearest Neighbours vs. Weighted k -Nearest Neighbour | 75 |
| 4.7.2.1 | Stand-alone k -NN and W- k -NN on the Leukaemia Dataset | 75 |
| 4.7.2.2 | Stand-alone k -Nearest Neighbours and Weighted k -Nearest Neighbour on the Prostate Cancer Dataset | 76 |
| 4.7.2.3 | Stand-alone k -Nearest Neighbours and Weighted k -Nearest Neighbour on the Ovarian Cancer Dataset | 76 |
| 4.7.3 | Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours vs. Two-Phase Evolutionary Algorithm/Weighted- k -Nearest Neighbours | 77 |
| 4.7.3.1 | End-Point Experiments for Phase I | 78 |
| 4.7.4 | The Set-Up of the Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm Using the Method Proposed in 4.7.3.1 | 82 |
| 4.7.5 | Application of Feature Weights | 88 |
| 4.7.6 | Conclusion | 94 |

5 Correlation Guided Evolutionary Algorithm/ k -Nearest Neighbours for Feature Selection 97

| | | |
|---------|---|-----|
| 5.1 | Introduction | 97 |
| 5.2 | Overview of the Algorithm | 100 |
| 5.3 | Correlation Guided Mutation | 101 |
| 5.3.1 | Correlation Guided Mutation - Remove a Gene | 102 |
| 5.3.1.1 | Preference Towards Selecting Strongly Correlated Features | 102 |
| 5.3.1.2 | Preference Towards Selecting Loosely Correlated Features | 103 |
| 5.3.2 | Correlation Guided Mutation - Add a Gene | 103 |
| 5.3.2.1 | Preference Towards Selecting Strongly Correlated Features | 103 |
| 5.3.2.2 | Preference Towards Selecting Loosely Correlated Features | 103 |
| 5.3.3 | Correlation Guided Mutation - Change a Gene | 104 |

| | | |
|----------|---|------------|
| 5.3.3.1 | Preference Towards Selecting Strongly Correlated Features | 104 |
| 5.3.3.2 | Preference Towards Selecting Loosely Correlated Features | 104 |
| 5.4 | Calculation of the Correlation Coefficient Matrix | 105 |
| 5.5 | Biased Random Number Generation | 106 |
| 5.6 | Experiment Set-up | 108 |
| 5.7 | Results and Discussion | 109 |
| 5.8 | Conclusion | 111 |
| 6 | Conclusions and Future Work | 117 |
| 6.1 | Main Findings | 117 |
| 6.1.1 | Parameters and Multi-Objective Approach in Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm | 117 |
| 6.1.2 | An Adaptive Weights Scheme for k -Nearest Neighbours in Evolutionary Algorithm/ k -Nearest Neighbours Algorithm for Feature Selection | 118 |
| 6.1.3 | Correlation Guided Evolutionary Algorithm/ k -Nearest Neighbours for Feature Selection | 119 |
| 6.2 | Future Work | 120 |
| 6.2.1 | Setting up of the Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm | 120 |
| 6.2.2 | Multi-objective Approach | 121 |
| 6.2.3 | Adaptive Weights | 121 |
| 6.2.4 | Correlation Guided Mutation for Evolutionary Algorithm/ k -Nearest Neighbours Algorithm | 123 |
| 6.3 | Final Thoughts | 124 |
| | Bibliography | 126 |

List of Tables

| | | |
|-----|---|----|
| 2.1 | Summary of results for various approaches to feature selection and classification in prostate cancer dataset. | 26 |
| 2.2 | Some well known multi-objective Genetic Algorithms adapted from Konak et al. [1] - part I. | 29 |
| 2.3 | Some well known multi-objective Genetic Algorithms adapted from Konak et al. [1] - part II. | 30 |
| 2.4 | Some well known multi-objective Genetic Algorithms adapted from Konak et al. [1] - part III. | 31 |
| 2.5 | Breast cancer dataset results by Sardana et al. [69]. The best performing classification method for each cross-validation technique is highlighted in bold face. | 32 |
| 3.1 | Effect of α on the objective function | 42 |
| 3.2 | Experiments run on the leukaemia dataset | 43 |
| 4.1 | Sample Dataset | 66 |
| 4.2 | Centroids | 66 |
| 4.3 | Calculating everything on the fly vs. look up for k -Nearest Neighbours | 75 |
| 4.4 | Results from the application of feature weights to the Evolutionary Algorithm/ k -Nearest Neighbours algorithm. | 89 |
| 4.5 | Results from the Evolutionary Algorithm/Weighted Centroid Classification algorithm. | 89 |
| 4.6 | Baseline results to compare the two weighted algorithms against. . . . | 89 |
| 4.7 | Analysis of the various ways to set-up the two-phase Evolutionary Algorithm/ k -Nearest Neighbours algorithm using Analysis Of Variance. Please refer to 4.7.4 for an explanation of each method shown in this table. | 90 |

| | | |
|------|--|-----|
| 4.8 | Mean classification accuracy and Standard Deviation for different ways of setting up the two-phase Evolutionary Algorithm/ k -Nearest Neighbours algorithm across five datasets. | 91 |
| 4.9 | Mean chromosome length and Standard Deviation for different ways of setting up the two-phase Evolutionary Algorithm/ k -Nearest Neighbours algorithm across five datasets. | 92 |
| 4.10 | Descriptives statistics on the results of various ways of setting up the two-phase Evolutionary Algorithm/ k -Nearest Neighbours algorithm across five datasets. | 93 |
| 4.11 | Calculation of mean ranks for each method of setting up two-phase Evolutionary Algorithm/ k -Nearest Neighbours algorithm. | 95 |
| 5.1 | An artificial dataset for illustrating the calculation of mean correlation coefficient for each feature. This dataset contains seven samples (S1 - S7). Each sample has six features F1 - F6). | 102 |
| 5.2 | Correlation coefficient matrix and the mean correlation coefficients for each feature for the dataset shown in 5.1. | 102 |
| 5.3 | Overall results for correlation guided mutation with preference for strongly correlated feature subsets for the Evolutionary Algorithm/ k -Nearest Neighbours algorithm. | 112 |
| 5.4 | Overall results for correlation guided mutation approach with preference for loosely correlated feature subsets for the Evolutionary Algorithm/ k -Nearest Neighbours algorithm. | 113 |
| 5.5 | Overall results for correlation guided mutation approach with preference for strongly correlated feature subsets for Evolutionary Algorithm/Weighted Centroid Classifier algorithm. | 114 |
| 5.6 | Comparison of correlation guided mutation approaches to other approaches used in this thesis. The best algorithm for each dataset is highlighted in boldface. | 115 |
| 6.1 | The best results from all the algorithms tested in this thesis. The best algorithm for each dataset is highlighted in boldface. | 125 |

List of Figures

| | | |
|-----|--|----|
| 2.1 | A Deoxyribonucleic Acid microarray (adapted from http://en.wikipedia.org/wiki/DNA) | |
| 2.2 | Taxonomy of feature selection techniques adapted from [68]. | 12 |
| 2.3 | Support Vector Machine (SVM) applied to the leukaemia dataset. Figure showing the application of the Support Vector Machine approach to the leukaemia dataset. Samples indicated by hollow triangle and rectangles are closest to the hyperplane and are therefore misclassified by the Support Vector Machine [80]. | 14 |
| 2.4 | k -Nearest Neighbours (k -NN) Technique. Figure illustrating k -Nearest Neighbours technique. Unknown type is classified as Type X using three closest neighbours. | 15 |
| 2.5 | Feature selection using an integer encoded chromosome. a) An example of a dataset that has 3 samples with 6 features in each sample. b) An integer encoded chromosome that encodes 3 features. c) The dataset after it has been reduced in size using the chromosome. . . . | 19 |
| 3.1 | The concept of dominated solutions. Solution 1 is non-dominated as solutions 2, 3 & 4 are worse than solution 1 on both objectives. Solutions 2 & 3 do not dominate each other as solution 2 is better than 3 on accuracy but solution 3 is better on length. Both 2 & 3 are dominated by solution 1. Solution 4 is dominated by solutions 1, 2 & 3. | 41 |

| | | |
|------|---|----|
| 3.2 | The effect of initial chromosome size on the value of the objective function. The value of the objective function of the initial randomly populated chromosomes decreases as the size of the chromosome is increased. This corresponds to an increase in the classification performance of these chromosomes. The value of the objective function during training stays low initially but increases as the size of the chromosome is increased corresponding to a degrading classification accuracy. The value of the objective function during testing follows the same trend as training albeit with markedly worse values due to over-fitting. | 45 |
| 3.3 | The effect of initial chromosome size on 3-fold cross-validated values of the objective function. Average cross-validated objective function values of chromosomes versus the initial size of the chromosome. . . . | 47 |
| 3.4 | The effect of α on the final length of chromosomes after 2000 generations. | 47 |
| 3.5 | Theoretical values produced by Equation 3.2 when the length of the chromosome is 30 and α is 1. | 48 |
| 3.6 | Theoretical values produced by Equation 3.2 when the length of the chromosome is 30 and α is 10, and when α is 1. | 49 |
| 3.7 | The effect that α has on final chromosomes on the ovarian cancer dataset after 2000 generations. | 49 |
| 3.8 | The effect that α has on the minimum, maximum and average length of final chromosomes on the ovarian cancer dataset after 2000 generations. | 50 |
| 3.9 | The effect that α has on the average length of chromosomes on the leukaemia dataset after 2000 generations. | 51 |
| 3.10 | The effect that α has on the classification accuracy on the leukaemia dataset after 2000 generations. | 52 |
| 3.11 | The effect that α has on the average length of chromosomes on the prostate cancer dataset after 2000 generations. | 52 |
| 3.12 | The effect that α has on the classification accuracy on the prostate cancer dataset after 2000 generations. | 53 |
| 3.13 | Best validated solutions for each experiment on the leukaemia dataset | 53 |

| | | |
|------|--|----|
| 3.14 | Best validated solutions for each experiment on the ovarian cancer dataset | 54 |
| 3.15 | Best validated solutions for each experiment on the prostate cancer dataset | 55 |
| 4.1 | Accuracy of stand-alone k -Nearest Neighbours on the leukaemia dataset for a range of values for T | 75 |
| 4.2 | Accuracy of stand-alone Weighted k -Nearest Neighbour on the prostate cancer dataset for a range of values for T | 76 |
| 4.3 | Accuracy of stand-alone Weighted k -Nearest Neighbour on the ovarian cancer dataset for a range of values for T | 77 |
| 4.4 | Leukaemia dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated taking the length of the chromosome into account and Binary Tournament Selection was used as the selection method. . . . | 80 |
| 4.5 | Leukaemia dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated without taking the length of the chromosome into account and Binary Tournament Selection was used as the selection method. . | 81 |
| 4.6 | Leukaemia dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated taking the length of the chromosome into account and Rank Based Selection was used as the selection method. | 82 |
| 4.7 | Leukaemia dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated without taking the length of the chromosome into account and Rank Based Selection was used as the selection method. | 83 |
| 4.8 | Ovarian cancer dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated taking the length of the chromosome into account and Binary Tournament Selection was used as the selection method. | 84 |

| | | |
|------|---|-----|
| 4.9 | Prostate cancer dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated taking the length of the chromosome into account and Binary Tournament Selection was used as the selection method. | 85 |
| 4.10 | The probability of a mean rank happening by random chance. | 88 |
| 4.11 | Comparison of the baseline method vs. weighted methods. | 94 |
| 5.1 | Frequency of 10000 biased random numbers with a <i>bias</i> of 0.5. | 107 |
| 5.2 | Frequency of 10000 biased random numbers with a <i>bias</i> of 0.25. | 107 |

Acronyms and Initialisms

***k*-NN** *k*-Nearest Neighbours.

ALH Adaptive Local Hyperplane.

ALL Acute Lymphoblastic Leukaemia.

AML Acute Myeloid Leukaemia.

ANN Adaptive Nearest Neighbour.

ANOVA Analysis Of Variance.

BTS Binary Tournament Selection.

cDNA complementary Deoxyribonucleic Acid.

cRNA complementary Ribonucleic Acid.

DMOEA Dynamic Multi-Objective Evolutionary Algorithm.

DNA Deoxyribonucleic Acid.

EA Evolutionary Algorithm.

EA/*k*-NN Evolutionary Algorithm/*k*-Nearest Neighbours.

EA/W-*k*-NN Evolutionary Algorithm/Weighted-*k*-Nearest Neighbours.

FS Feature Selection.

GA Genetic Algorithm.

HKNN *k*-local Hyperplane Distance Nearest Neighbour.

IFTRSM Incremental Formulation of Trace of Ratio of Scatter Matrices.

JVM Java Virtual Machine.

KSVM k -Nearest Neighbours & Support Vector Machine Classifier.

LDC Linear Discriminant Classifier.

LOOCV Leave One Out Cross Validation.

MOEA Multi-Objective Evolutionary Algorithm.

MOGA Multi-Objective Genetic Algorithm.

MOP Multi-objective Optimisation Problem.

mRMR minimum Redundancy and Maximum Relevance.

NPGA Niched Pareto Genetic Algorithm.

NSGA Nondominated Sorting Genetic Algorithm.

NSGA-II Nondominated Sorting Genetic Algorithm-II.

PAES Pareto Archived Evolution Strategy.

PESA Pareto Envelope-based Selection Algorithm.

RAM Random Access Memory.

RBS Rank Based Selection.

RDGA Rank Density-based Genetic Algorithm.

RFE Recursive Feature Elimination.

RWGA Random Weighted Genetic Algorithm.

S2N Signal-to-Noise Metric.

SD Standard Deviation.

SELDI-TOF Surface-Enhanced Laser Desorption and Ionisation Time-Of-Flight.

SPEA Strength Pareto Evolutionary Algorithm.

SPEA-2 Strength Pareto Evolutionary Algorithm-2.

SVM Support Vector Machine.

VEGA Vector Evaluated Genetic Algorithm.

W- k -NN Weighted k -Nearest Neighbour.

WBGA Weight-Based Genetic Algorithm.

Chapter 1

Introduction

1.1 Introduction

Recent advances in technology have tremendously increased humankind's ability to create, store and carry out computations on digital data. It is estimated that in 2007, humankind had the capacity to store 44716 MB of optimally compressed data [29] per capita. In 1986, this figure was 539 MB [29]. This represents roughly an 83-fold increase in the per capita capacity to store data. Humankind's capacity for carrying out computations on digital data has seen an even bigger increase. Hilbert et al. [29] estimated that in 1986, humankind had the per capita capacity to carry out 0.06 Mega Instructions Per Second (MIPS) on general purpose computers. By 2007, this figure had increased to 968 MIPS. In other words, this represents roughly a 16133-fold increase in the per capita capacity to carry out computations using general purpose computers in 21 years (1986 to 2007).

This explosion in both storage and computational capacity together with other scientific and technological advances has inevitably led to large scientific datasets. This is especially the case in the biomedical field. Recent advances in technologies such as high density Deoxyribonucleic Acid (DNA) microarrays and Proteomic analysis has led to modern biomedical datasets that are usually rich in features. For example, the ovarian cancer dataset introduced by Petricoin et al. [61] has 15154 attributes (features) per sample. It is believed that only a handful of features are significantly differently expressed in a disease sample compared to a normal sample. This means that data from DNA microarrays or Proteomic analysis is highly redundant and contain high dimensional noise [38, 46].

These datasets are valuable in that they have the potential to help us understand the pathology of certain diseases. A deeper understanding of the pathology of diseases will help us deal better with these diseases. Another use of these datasets is that machine learning can be used to create models that help us in early diagnosis of certain diseases. However, the number of features present in a typical bioscience dataset presents a formidable barrier to such efforts.

This and other challenges associated with extracting high-level knowledge from real life large datasets has led to the emergence of the field of predictive data mining [79]. In this context, the aim of Feature Selection (FS) is to eliminate features that seem irrelevant to the case under study. FS on a feature-rich dataset results in a much reduced dataset. Predictive data mining can then perform faster with increased accuracy on these reduced datasets.

FS is in itself a rich area of research, and many of the techniques in this area rely on the use of statistical correlation measures to rank the features. However, though a convenient and fast approach to FS and very often used, statistical ranking based FS can be unwise; such methods will miss non-linear interactions between features, which in turn may be common in many datasets of interest.

1.1.1 Feature Selection and Classification

In predictive data mining, a model can be created that consists of a subset of features of the original dataset. In case of gene expression data related to a certain disease (e.g. cancer), it is possible that this subset of features can then be used as a predictor in unseen samples leading to early diagnosis. Therefore, FS is a technique commonly used in predictive data mining for building robust learning models that can be used to classify hitherto unseen data.

As the number of profiled features increase, the number of possible feature subsets that may be of importance grow exponentially. This makes the use of exhaustive search for FS infeasible.

This leaves heuristic search techniques such as Evolutionary Algorithms (EAs) as prime candidates for selecting feature subsets that are capable of discriminating between disease and normal cases [38, 50]. Many researchers are concentrating their efforts on methods that combine advanced search techniques (e.g. EAs) together with efficient classification techniques (e.g. k -Nearest Neighbours (k -NN)) for feature

selection and classification [38].

The most important objectives of feature selection are [68]:

- to avoid over-fitting and improve model performance, i.e. prediction performance of selected feature subset on unseen data.
- to provide faster and more cost-effective models.
- to gain a deeper insight into the underlying processes that generated the data.

1.2 Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm for Feature Selection and Classification

Juliusdottir et al. [38] has introduced a two-phase Evolutionary Algorithm/ k -Nearest Neighbours (EA/ k -NN) algorithm for FS and classification in DNA microarray datasets that is capable of achieving comparable, if not better results, compared to other methodologies.

The two-phase EA/ k -NN algorithm runs the EA/ k -NN algorithm as both the prior-selection stage (on the complete dataset) and machine learning stage (on a reduced dataset). The EA used in the EA/ k -NN algorithm is a generational, elitist EA that uses k -NN as the classifier/objective function. In the objective function, the classification error and the length of the chromosome are combined together to form a single metric that is minimised.

They argue that the use of k -NN as a classifier puts the onus on the search technique to find salient and significant gene subsets. Therefore, the two-phase EA/ k -NN algorithm is capable of finding significant gene subsets which may have been overlooked with a more efficient classifier than k -NN.

Another advantage of using the EA/ k -NN algorithm for initial feature selection (phase one) is that owing to generally good classification performance of k -NN, we can be confident that a good subset of features are selected. In other words, the chances of discarding a significant gene or a set of significant genes is lower when using this method compared to other methods.

Their two-phase EA/ k -NN algorithm has led directly to the identification of three genes for prostate cancer and five genes for colon cancer. This supports previous

studies and further strengthens findings from those studies and suggests good targets for further research by domain experts.

Furthermore, two-phase EA/ k -NN algorithm is relatively straightforward to implement and runs at an acceptable speed for even large datasets. However, there are areas in which the two-phase EA/ k -NN algorithm could be better optimised and configured. Therefore, it was decided that this thesis would consist of a thorough investigation of the two-phase EA/ k -NN algorithm as a candidate for FS and classification in predictive data mining in large biomedical datasets.

In particular, this thesis investigated the optimal way to set up the two-phase EA/ k -NN algorithm so that it performs well across multiple datasets. This included an investigation into the population size, initial chromosome size, the balance between classification accuracy and the length of the chromosome in the objective function, number of generations to run phase one and phase two of the algorithm for and different ways in which genes can be selected during phase one that can then form the starting point of phase two. As an alternative to tuning some of the parameters of the algorithm (e.g. the balance between classification accuracy and the length of the chromosome in the objective function), an investigation into a multi-objective two-phase EA/ k -NN was also carried out.

1.3 Datasets

1.3.1 Leukaemia Dataset

This is a publicly available dataset introduced by Golub et al. [22]. Their initial leukaemia dataset consisted of 38 bone marrow samples (27 Acute Lymphoblastic Leukaemia (ALL) samples & 11 Acute Myeloid Leukaemia (AML) samples) each containing 7070 genes. They tested their results on an independent dataset that had 34 samples (20 ALL, 14 AML).

These two datasets were mixed together to form a single dataset that had 72 samples out of which 47 samples were ALL and 25 samples were AML. The challenge with this dataset is to build models that can effectively classify a sample as either ALL or AML.

1.3.2 Ovarian Cancer Dataset

The ovarian cancer dataset is a publicly available dataset (Proteomic analysis data) introduced by Petricoin et al. [61]. This dataset contains 253 samples of which 91 are normal and 162 are cancer. Each sample contains 15154 values (features or genes). The aim of predictive data mining in this case is to classify unseen samples either as cancer or normal.

1.3.3 Prostate Cancer Dataset

This dataset was introduced by Singh et al. [72]. It contains 52 tumour and 50 normal samples with 12600 features per sample. As with the ovarian cancer dataset, the aim in this case is to predict whether an unseen sample is a cancer sample or normal sample.

1.3.4 Breast Cancer Dataset

The breast cancer dataset was introduced by Van't Veer et al. [77] in patient outcome prediction for breast cancer. The original dataset was divided into training and test datasets. For the purpose of this thesis, both training and test datasets were mixed together to form one dataset that was then randomly split into smaller datasets as required. The complete dataset contains 46 samples of “relapse” cases and 51 “non-relapse” samples. Each sample contains 24481 genes. The aim of predictive data mining in this case is to predict the patient outcome as either “relapse” or “non-relapse”.

1.3.5 Colon Cancer Dataset

This dataset contains 62 samples collected from colon cancer patients and was introduced by Alon et al. [2]. It contains 40 tumour samples and 22 normal samples taken from a healthy part of the colon from the same patients. Each sample contains 2000 genes.

In their original study, Alon et al. [2] studied gene expression patterns using Affymetrix oligonucleotide arrays complementary to more than 6500 genes. However, they then selected 2000 genes based on the confidence levels of the measurements of gene expression and used only these 2000 genes in their final analysis. Therefore,

it was decided to use the same 2000 genes in this thesis. The aim of predictive data mining in this case is to predictively discriminate between tumour and healthy samples.

1.4 Contributions

The following is a list of contributions made by this thesis to the field of FS and classification in predictive data mining. In particular, this thesis deals with large biomedical datasets.

- Juliusdottir et al. [38] introduced a novel two-phase EA/ k -NN algorithm for FS and classification in DNA microarray datasets. The first contribution of this thesis is the investigation of the setting up of the two phases in the two-phase EA/ k -NN algorithm including the parameters that needed to be tuned. This investigation was based on the hypothesis that phase one of the algorithm is critical to the success of the algorithm as only the genes selected during phase one are used for model building in phase two. The investigation revealed that some of the parameters need to be tuned correctly for each dataset for the algorithm to perform as described by Juliusdottir et al. [38].
- As an alternative to tuning parameters, a multi-objective EA was proposed in this thesis that could replace the single objective EA in the two-phase EA/ k -NN algorithm. The multi-objective algorithm simultaneously optimises both the length of the chromosome (the number of features in the selected subset) and the classification accuracy without requiring a pre-tuned parameter labelled α . The multi-objective approach yielded very competitive results compared to the two-phase EA/ k -NN algorithm.
- An investigation was carried into applying adaptive weights (adopted from Yang and Kecman [81]) to the k -NN algorithm in order to determine if weighted k -NN (Weighted k -Nearest Neighbour (W- k -NN)) would lead to discovery of optimal feature subsets. As with α in the single objective two-phase EA/ k -NN algorithm, there are a few parameters that needs to be pre-tuned for W- k -NN to perform as expected. This thesis contributes that, with proper tuning, W- k -NN is able to out-perform k -NN.

- Another contribution made by this thesis is the introduction of a novel weighted centroid classification technique as the objective function for the EA in the combined EA/ k -NN approach. With this classification technique, the EA weighted centroid classification algorithm is able to out-perform both the EA/ k -NN algorithm and the Evolutionary Algorithm/Weighted- k -Nearest Neighbours (EA/W- k -NN) algorithm.
- Classical statistical techniques (e.g. Analysis Of Variance (ANOVA)) were found to be ineffective at comparing multiple algorithms across multiple datasets in order to determine the best algorithm across all the datasets. In order to overcome this problem, a randomisation statistics approach was investigated and adopted in this thesis.
- Finally, this thesis introduces a correlation guided mutation operator. This mutation operator is designed towards selecting highly correlated features with a higher probability of being included in the chromosome. The results indicate this technique to be a promising technique for FS and classification.

1.5 Publications Resulting From this Research

Manjula SB Dissanayake and David W Corne. Feature selection and classification in bioscience/medical datasets: Study of parameters and multi-objective approach in two-phase EA/ k -NN method. *Computational Intelligence (UKCI), 2010 UK Workshop on*. IEEE, 2010.

1.6 Outline of Thesis

This thesis is organised as follows:

- Chapter 2 provides a review of FS and classification techniques found in literature. It pays particular attention to FS techniques for DNA microarray data. It also contains a detailed introduction to the two-phase EA/ k -NN algorithm and the adaptive weights scheme.
- Chapter 3 presents a detailed investigation into the configuration of the two phases of the two-phase EA/ k -NN algorithm. It also presents an investigation into a multi-objective approach for FS and classification.

- Chapter 4 takes the investigation of the two phases started in Chapter 3 further. This chapter also introduces an adaptive weights scheme for k -NN algorithm and a novel weighted centroid classification technique.
- Chapter 5 provides an investigation into a novel correlation guided mutation operator for the EA/ k -NN algorithm.
- Chapter 6 then presents a summary of the conclusions made in this thesis and presents a discussion of promising areas of further research.

Chapter 2

Literature Review

2.1 Deoxyribonucleic Acid Microarrays

DNA microarrays, shown in Figure 2.1, are a relatively new, sophisticated technology used in molecular biology and medicine. They were first introduced in 1994 by Pease et al. [59].

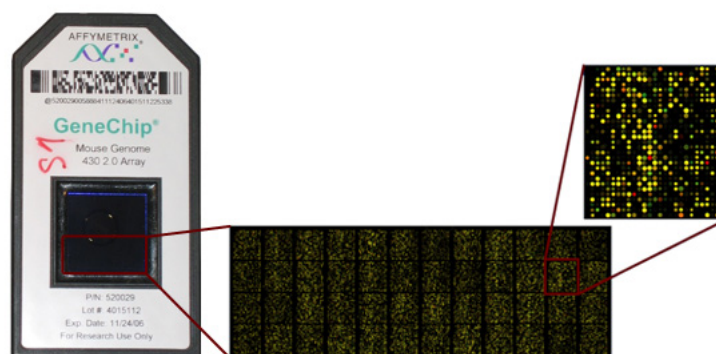


Figure 2.1: A Deoxyribonucleic Acid microarray (adapted from http://en.wikipedia.org/wiki/DNA_microarray)

Each DNA microarray consists of thousands of microscopic wells of DNA oligonucleotides arranged into a two-dimensional (2D) array shape. Each well contains picomoles of a specific DNA sequence. These wells in a DNA microarray are called features.

Each feature is capable of binding its corresponding complementary Deoxyribonucleic Acid (cDNA) or complementary Ribonucleic Acid (cRNA) sequences. These cDNA or cRNA sequences are called targets. Targets are labelled with fluorophores or luminescence chemicals. It is possible to measure the amount of a target bound to each feature by measuring the amount of fluorescence [37].

DNA microarrays can be used to measure changes in expression levels of many thousands of genes simultaneously. This ability makes them a very powerful tool that can be used in early diagnosis and treatment discovery for many diseases [38].

The standard method for isolating genes responsible for a certain disease (e.g. cancer) is to measure gene expression levels of a number of patients (e.g. a couple of hundred) and compare them with expression levels of the normal population. As each chip is capable of monitoring many thousands of genes at one time and as the data collected from these experiments tend to be very noisy, this opens up a new challenge for computer scientists: feature selection and classification [38].

Typically, a microarray dataset consists of many thousands of genes but relatively few samples (a couple of hundred). This means that there may be many subsets of genes with good classification performance. The aim of feature selection and classification is to find as many near-optimal solutions as possible. The most frequently selected genes in these subsets can then be studied further as they have a better chance of being significant to the case under study [48].

2.2 Overview of Feature Selection and Classification Techniques

As explained in 1.1.1, exhaustive searches for subsets of interesting features become infeasible as datasets get larger. This is due to the fact that if the original dataset contained N number of features, then the total number of possible feature subsets is 2^N [13]. Therefore, even for relatively small datasets, complete or exhaustive search for the optimal feature subset becomes infeasible. This leaves heuristic search techniques such as EAs as prime candidates for selecting feature subsets that are capable of discriminating between disease and normal cases [38, 50] in large biomedical datasets.

Feature selection can be divided into supervised learning (e.g. classification where the class value is known in advance) and unsupervised learning (e.g. clustering) [68]. As this thesis studies feature selection and classification in large biological datasets where the training data always contain class values, the appropriate learning method for feature selection for this thesis is supervised learning.

A taxonomy of feature selection techniques applicable to large biomedical datasets

is shown in Figure 2.2. Feature selection and classification techniques can be divided into four categories [68]:

- Filter
- Wrapper
- Embedded
- Hybrid

2.2.1 Filter Techniques

These techniques work by looking at the intrinsic properties of the dataset. For example, in most cases, a feature relevance score is calculated and low scoring features are pruned. The remainder of the dataset is then used for classification. The advantages of these techniques are that they can be easily scaled up or down, they are computationally simple and they are fast.

However, there are many disadvantages to this approach. For example, these techniques prune the dataset by looking at one feature at a time. This means that inter-feature relationships are not taken into account.

A number of multivariate filter techniques (e.g. Markov blanket filter) have been introduced in order to address some of the issues associated with univariate filter techniques [68].

Clustering analysis is also widely used with microarray data [2] for feature selection and classification. Clustering analysis works by looking at correlation between groups of genes and provides insight into gene-gene interaction. However, clustering analysis is not well suited for classification as it looks at correlated patterns of expression rather than patterns of expression that can differentiate between samples. It is also difficult to determine the relative importance of genes by using clustering analysis [48].

2.2.2 Wrapper Techniques

In wrapper techniques, a search procedure is defined that is capable of searching through the space of possible feature subsets for a given dataset. The search procedure generates various subsets of features and they are evaluated against a test

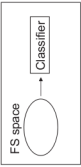


| | Model search | Advantages | Disadvantages | Examples |
|----------|---|--|--|---|
| Filter |  | Fast Scalable Independent of the classifier | Ignores feature dependencies | Chi-square Euclidean distance t-test |
| | | Univariate | Ignores interaction with the classifier | Information gain, Gain ratio [6] |
| Wrapper |  | Models feature dependencies Independent of the classifier Better computational complexity than wrapper methods | Slower than univariate techniques Less scalable than univariate techniques Ignores interaction with the classifier | Correlation based feature selection (CFS) [45] Markov blanket filter (MBF) [62] Fast correlation based feature selection (FCBF) [136] |
| | | Multivariate | Risk of over fitting More prone than randomized algorithms to getting stuck in a local optimum (greedy search) | |
| | | Deterministic | Classifier dependent selection | Sequential forward selection (SFS) [60] Sequential backward elimination (SBE) [60] Plus q take-away r [33] Beam search [106] |
| | | Randomized | Computationally intensive Classifier dependent selection Higher risk of overfitting than deterministic algorithms | Simulated annealing Randomized hill climbing [110] Genetic algorithms [50] Estimation of distribution algorithms [52] |
| Embedded |  | Interacts with the classifier Better computational complexity than wrapper methods Models feature dependencies | Classifier dependent selection | Decision trees Weighted naive Bayes [28] Feature selection using the weight vector of SVM [44, 125] |

Figure 2.2: Taxonomy of feature selection techniques adapted from [68].

set by the classification algorithm [38, 68]. Therefore these techniques are tailored to whatever the classification algorithm that is used in the evaluation. Wrapper techniques are able to take inter-feature relations into account. However, they can be very computationally expensive, especially if the classification step is computationally heavy [68].

These techniques are called “wrapper” techniques [38, 68] as the search process is “wrapped” around the classification model, enabling the search process to search for more efficient classification models.

In large datasets, the search algorithms used in wrapper techniques tend to be heuristic search algorithms such as genetic algorithms. This, as explained in 1.1.1, is due to the fact that exhaustive search is infeasible in large datasets.

2.2.3 Embedded Techniques

This class of feature selection techniques is termed “embedded techniques” as the search for an optimal subset of features is built into the classifier construction [68]. Decision trees are an example of an embedded technique used in feature selection.

Embedded techniques are also specific to a given classification (learning) algorithm. However, embedded techniques are less computationally intensive than wrapper techniques.

2.2.4 Hybrid Techniques

Hybrid techniques combine two techniques to obtain better performance in FS. k -Nearest Neighbours & Support Vector Machine Classifier (KSVM) proposed by Xiaoqiao and Lin [80] belongs to this category. KSVM is a new classifier that combines Support Vector Machine (SVM) together with k -NN. In the classification phase, the algorithm computes the distance from test samples to the optimal hyperplane of SVM in feature space. If the distance is greater than a given threshold, then the test sample will be classified on the SVM, otherwise k -NN will be used for classification.

Xiaoqiao and Lin [80] explain SVM as “a method for finding a hyperplane in high dimensional space that separates training samples of each class while maximizing the minimum distance between that hyperplane and any training sample. If the data are not linearly separable, they can be projected onto a higher dimensional

feature space in which they are separable”.

The SVM works by identifying training samples that are closest to the hyperplane and classifies an unseen sample based on this information. However, the performance of the SVM degrades when there are training samples which are very close to the hyperplane. This is illustrated in Figure 2.3.

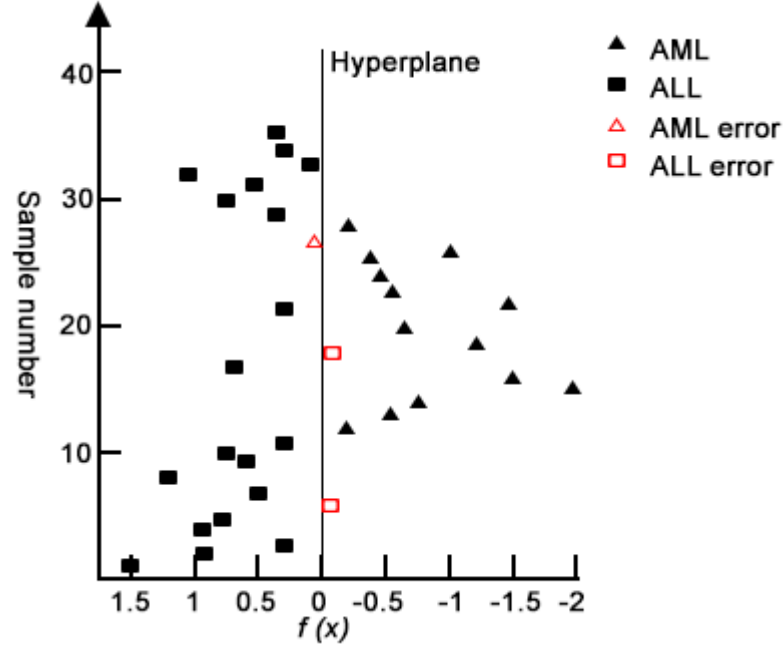


Figure 2.3: Support Vector Machine (SVM) applied to the leukaemia dataset. Figure showing the application of the Support Vector Machine approach to the leukaemia dataset. Samples indicated by hollow triangle and rectangles are closest to the hyperplane and are therefore misclassified by the Support Vector Machine [80].

k -NN works by looking at distance between an unknown test sample and known training samples and classifying the unknown sample according to the majority of its closest neighbours.

Xiaoqiao and Lin [80] used Signal-to-Noise Metric (S2N) for feature selection and used KSVM for classification.

Their results indicate that KSVM has better accuracy than either k -NN or SVM alone. They conclude that this may be due to the fact that KSVM gathers more support vectors during training and therefore carries more information.

Furthermore, the number of genes used in the training process has less effect on KSVM as opposed to SVM also due to the fact that KSVM carries more information.

Mei et al. [52] also propose a similar hybridized k -NN-SVM approach. However, they only use an SVM for classification when k -NN classification is indecisive.

2.3 Evolutionary Algorithm/ k -Nearest Neighbours Algorithm

The EA/ k -NN algorithm uses an EA as the search technique and k -NN algorithm as the classifier. The EA/ k -NN algorithm was first reported by Siedlecki & Skalan-sky [71].

k -NN works by assigning a classification to a sample based on its closest neighbours.

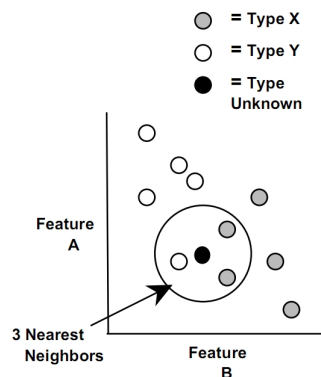


Figure 2.4: k -Nearest Neighbours (k -NN) Technique. Figure illustrating k -Nearest Neighbours technique. Unknown type is classified as Type X using three closest neighbours.

In Figure 2.4, it is assumed that there are only two features (A & B). They are assigned to x axis and y axis. Samples are then placed in this space using their known values. A sample of unknown type can then be placed in this space by its feature values. This unknown sample can then be classified by looking at its k -NN. In Figure 2.4, unknown type can be classified as type X by looking at its 3 closest neighbours [62].

EA/ k -NN algorithm was applied to Surface-Enhanced Laser Desorption and Ionisation Time-Of-Flight (SELDI-TOF) Proteomic data by Li et al. [47]. SELDI-TOF data is usually more complicated than DNA microarray data as SELDI-TOF tends to contain more samples (hundreds of samples) and more features for each sample [47].

Li et al. [47] applied EA/ k -NN algorithm to find many near optimal feature subsets. Features were then ranked by frequency of occurrence and the most frequently occurring features were used to classify unseen data.

They concluded that EA/ k -NN algorithm was able to find a subset of 10 features that was able to classify optimally between cancer and non-cancer cases in the

ovarian cancer SELDI-TOF Proteomic dataset.

Therefore, it is apparent that EA/ k -NN algorithm is suited for feature selection in predictive data mining.

2.4 Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm

k -NN algorithm, first introduced by Fix & Hodges [18] is a fast and simple algorithm with the advantage of having good classification performance on a wide range of real world datasets [38, 64].

Siedlecki & Sklansky [38, 71], who first introduced the idea of combining an EA with k -NN algorithm, showed that a combined EA/ k -NN is very efficient at finding near optimal subsets of features from a large dataset.

Jirapech-Umpai & Aitken [38, 33] showed that classification performance of selected subsets of features improved significantly when prior feature selection was employed. They showed this by applying EA/ k -NN algorithm without prior feature selection to Golub's leukaemia dataset [38, 22].

They used chromosomes with initial size set to 10 and a small population. This produced 68% accuracy at best on the test set, which is poor. The EA converged quickly due to the fact that the risk of getting stuck in local optima is very high with a large dataset and small chromosome size [38].

They then used RankGene software for initial feature selection. EA/ k -NN algorithm was then applied to the 100 best genes from the initial feature selection phase. This resulted in 95% accuracy on the test set which is a very significant increase from 68%. This clearly showed that EA/ k -NN performs better when applied to a reduced dataset after prior feature selection.

Stochastic search methods such as EAs return different results for different runs when applied to truly complex problems [38]. Although this could be considered an unfavourable outcome under certain circumstances, in the case of feature selection and classification, it is desirable to get different results for different runs. This is because the search method could be run multiple times and results could be pooled to produce a reduced yet diverse dataset to which further selection and classification methods could be applied.

It is also favourable as it is possible to run either the further selection & classification step or the whole process multiple times to obtain a diverse set of near optimal feature subsets. Then, it is possible to analyse frequently occurring genes within these subsets and conclude, with some confidence, that these genes are significant to the case under study [38].

Juliusdottir et al. [38], in taking this idea forward, decided to use EA/ k -NN for initial feature selection as well as further selection and classification. During the first phase, the algorithm was applied to the whole dataset for feature selection. This enabled them to reduce the datasets down to smaller sizes and apply the same algorithm to these smaller datasets. Unlike filter methods, this approach has the benefit of good feature discovery without initial dimensionality reduction. They showed that the application of EA/ k -NN algorithm as the pre-processing method (phase one) is capable of competitive, if not better results, compared to other more complex pre-processing methods.

It has been shown that problems with unimodal fitness landscapes where there is only one isolated global optimum with little or no information available elsewhere in the landscape are difficult to solve. Problems with such isolated peaks in the landscape have been called needle-in-a-haystack (NIAH) problems. It has been shown that in order to make such a problem solvable by Genetic Algorithms (GAs), the fitness landscape has to be modified to decrease the isolation of the single optimum and to increase its basin of attraction [30, 7]. On the other hand, in the context of feature selection, a generally flat fitness landscape would also prevent an EA from selecting a set of genes that are relevant to the case under study.

In a feature selection and classification context, a highly sophisticated classifier such as an SVM may contribute to the flattening of the fitness landscape and therefore decrease the amount of useful information that is available to the EA. For example, an SVM may classify a sub-optimal feature subset with 95% accuracy. k -NN on the other hand may classify the same subset with 80% accuracy. As the classification accuracy is measured as a percentage, the maximum possible accuracy is 100%. An SVM will therefore reduce the gap between this sub-optimal feature subset and the optimal feature subset leading to a flattening of the fitness landscape. k -NN on the other hand will not flatten the landscape to the same extent as an SVM. Therefore, k -NN will guide the EA towards optimal feature subsets. Juliusdottir et al. [38] argue that, in a combined feature selection/classification context, it is highly

valuable to concentrate on classification methods that are straightforward as they will put the onus on the EA to search for subsets of genes that are strongly correlated to the case under study. Only then does it become possible to save money and time by letting domain experts concentrate on these genes to either find a cure for the underlying disease or come up with early diagnostic tests, which is the ultimate goal of feature selection classification on these datasets.

2.4.1 Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm Explained

Juliusdottir et al. [38] configured phase one of the two-phase EA/ k -NN as described below:

- Chromosome length = 200
- Population = 30
- Generations = 500
- Selection type = Roulette wheel selection
- Elitism = Yes, elite count of 2
- Mutation rate = 0.3 (30% chance of mutating to a non-zero gene, 70% chance of removing a gene)
- Crossover = Single point crossover
- Number of neighbours (k) = 3

Figure 2.5 shows how a chromosome encodes a subset of features from a dataset (adapted from Juliusdottir et al. [38]).

In Figure 2.5:

- a) S1, S2, S3 represents three samples from a dataset. F1 - F6 represents values for each feature in a sample.
- b) An integer encoded chromosome. This chromosome is encoding features 2, 3 & 5 from the initial dataset. 2, 3 & 5 can be referred to as “selected feature subset”.

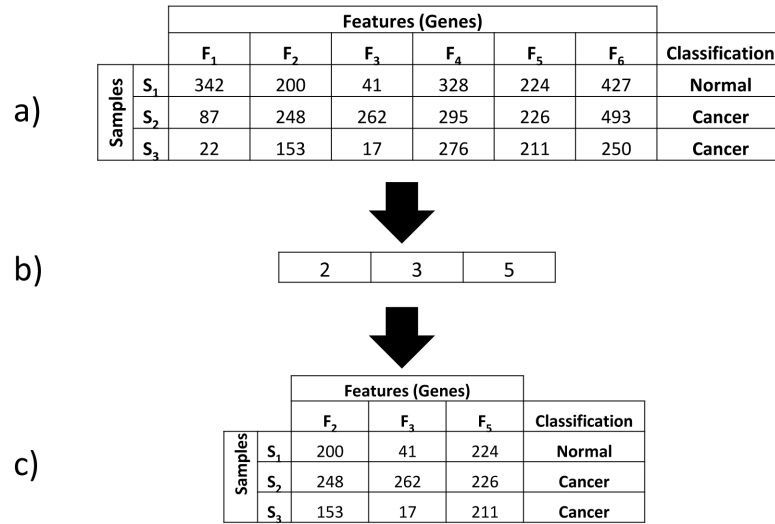


Figure 2.5: Feature selection using an integer encoded chromosome. a) An example of a dataset that has 3 samples with 6 features in each sample. b) An integer encoded chromosome that encodes 3 features. c) The dataset after it has been reduced in size using the chromosome.

c) This table shows how the dataset can be reduced using the chromosome shown in b).

Juliusdottir et al. [38] chose to represent a subset of features (i.e. a chromosome) as a variable length list of integers as shown in Figure 2.5. They argued that this encoding had the benefit of limiting *a priori* the size of a feature subset. They also argued that this approach had the added benefit of scalability. For large datasets (e.g. microarray data), a binary chromosome would need to contain many thousands of bits as the number of bits need to be equal to the number of attributes in the dataset.

Algorithms used in this thesis have been implemented using Java. In Java, the most efficient way of representing a binary chromosome is to represent it as an array of bytes. A byte is a primitive data type in Java that takes up exactly 1 byte of memory [23]. As the number of bits in a binary encoded chromosome needs to be equal to the number of attributes in a dataset, a binary chromosome represented in memory as a byte array will take memory equal to the number of attributes in the dataset in bytes. For example, the ovarian cancer dataset used in this thesis consists of 15154 attributes. A binary encoded chromosome can be implemented in Java using a byte array that can hold 15154 bytes. This array would take 15154 bytes of memory (excluding the overhead imposed by the Java Virtual Machine (JVM)).

An integer encoded chromosome can be implemented as an array of integers in Java. An integer takes up 4 bytes of memory in Java [23]. If an integer encoded chromosome is to represent the entire feature set (worst case scenario) of the dataset, then for the ovarian dataset, the chromosome would take up 60616 bytes (15154 x 4 bytes) of memory. However, the initial chromosome is limited to 400 features. Therefore, the initial memory imprint would be limited to 1600 bytes (400 x 4 bytes) per chromosome. One of the objectives of the algorithm is to minimise the size of the feature subset encoded by the chromosome. Therefore, most chromosomes in the population are likely to be reduced to a few features. As shown above, the shorter the chromosome, the more efficient integer encoding becomes.

There is also evidence in the literature to suggest that integer or floating point representation of a chromosome is a faster and more consistent form to run [32]. Also, an integer encoding is obvious, easy to decode and meaningful crossover and mutation operators can be applied with relative ease [12].

Fitness of a given chromosome is calculated using the Equation 2.1.

$$Fitness = ((100 - class_acc)/100) + ((n/N/\alpha) \quad (2.1)$$

Where:

- *class_acc* = mean classification performance over the three three-fold cross-validation runs
- *n* = size of chromosome
- *N* = maximum possible length for a chromosome
- α = parameter controlling trade-off between preference for accuracy and preference for small subset sizes

Classification accuracy (*class_acc*) for a given chromosome is calculated by looking at how many samples in the dataset that it is able to classify accurately using *k*-NN algorithm.

To determine if a given chromosome (x) is capable of classifying a given sample (s1) in the dataset:

- Calculate the Euclidean distances to all the other samples in the dataset using

features encoded in x . E.g. for s_1 , calculate distance from s_1 to s_2 , s_1 to s_3 and so on.

- Pick three closest neighbours to s_1 from these distances.
- Determine the class of the majority of neighbours of s_1 . Assign this class to s_1 .
- Compare the known class of s_1 with the assigned class. If they match, then x is capable of classifying s_1 correctly using the k -NN algorithm.

The above procedure is repeated for all the data samples for a given chromosome and number of data samples that the chromosome is able to classify accurately is counted. Classification accuracy is then calculated as number of correct classifications/total data samples. Once classification accuracy is known, the fitness for a given chromosome can be calculated using the Equation 2.1.

2.4.2 The Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Experimental Design

In this thesis, the two-phase EA/ k -NN algorithm introduced by Juliusdottir et al. [38] is used as the baseline. In Chapter 3, a thorough investigation is carried out into the optimal way of arranging the two-phase EA/ k -NN algorithm so that it performs well across a range of datasets. In Chapter 4, a weighting scheme for k -NN is investigated and a novel weighted centroid classifier is introduced in place of k -NN. Chapter 5 investigates a correlation guided mutation operator for the EA/ k -NN algorithm. As the experimental design used by Juliusdottir et al. [38] is used as the basis for the following chapters, a detailed explanation of the experimental design is given in this section.

Juliusdottir et al. [38] conducted all their experiments in two phases. EA/ k -NN algorithm was run on two datasets (colon [2]; prostate [72]) repeatedly during phase one. Genes that appeared in final populations were pooled together to create two large but much reduced datasets from the original datasets. EA/ k -NN was then run on these reduced datasets for further selection and classification.

2.4.2.1 Phase One

Experiment 1A

This experiment was carried out on the prostate cancer dataset. The dataset had:

- 12600 features
- 52 prostate cancer samples
- 50 normal samples

The dataset was divided into training and test sets. The training set had 39 cancer and 36 normal samples. The remaining 27 samples were used as the validation set.

Set-up of the EA:

- Total number of runs = 10
- Generations = 400
- Chromosome length = 400
- Population size: 80
- Elite count: 2
- Selection: Roulette wheel selection
- Crossover: Single point crossover
- Number of neighbours for k -NN: 3

After running EA/ k -NN for ten runs, the final best subsets were pooled together to form a dataset that contained 245 unique genes. This dataset was then used as the starting point for experiment 1B.

Experiment 2A

This experiment was carried out on the colon cancer dataset. The dataset had:

- 2000 features
- 40 colon cancer samples
- 22 normal samples

The dataset was divided into 4 subsets of similar sizes. Three of these were used for 3-fold cross-validation (2 training, 1 testing) while the other was used as the validation set.

Set-up of the EA:

- Total number of runs = 10
- Generations = 400
- Chromosome length = 200
- Population size: 30
- Elite count: 2
- Selection: Roulette wheel selection
- Crossover: Single point crossover
- Number of neighbours for k -NN: 3

After running EA/ k -NN for ten runs, the final best subsets were pooled together to form a dataset that contained 151 unique genes. This dataset was then used as the starting point for experiment 2B.

2.4.2.2 Phase Two

Experiment 1B

This experiment was run using 245 genes discovered from experiment 1A in phase one.

Set-up of the EA:

- Total number of runs = 10
- Generations = 100
- Chromosome length = 100
- Population size: 30
- Elite count: 2

- Selection: Roulette wheel selection
- Crossover: Single point crossover
- Number of neighbours for k -NN: 3

The final best subsets were pooled to form a best subset of size 20.

Experiment 2B

This experiment was run using 151 genes discovered from experiment 2A in phase one.

Set-up of the EA:

- Total number of runs = 8
- Generations = 100
- Chromosome length = 70
- Population size: 30
- Elite count: 2
- Selection: Roulette wheel selection
- Crossover: Single point crossover
- Number of neighbours for k -NN: 3

The final best subsets were pooled to form a best subset of size 37.

2.4.3 Results from Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm

Using a Probabilistic Model Building Genetic Algorithm(PMBGA) with Support Vector Machine as a classifier, Topon & Iba [38, 58] managed to collect 177 different subsets. Out of these, the best subsets returned a test set accuracy of 94.12%. This subset included 24 unique genes. The smallest subset they obtained contained only 6 genes and returned 82.35% testing accuracy. They obtained an average of 84.29% \pm 4.57 test set accuracy with the average number of selected genes being 17.14 \pm 7.4.

Singh et al. [72] used signal/noise stats together with k -NN to obtain a 16 gene subset that returned 93.12% training accuracy.

Juliusdottir et al. [38] managed to obtain 10 subsets including an average of 28 genes after applying EA/ k -NN algorithm once for the whole dataset. The average test set accuracy was 87.04%. This improved to 88.88% when EA/ k -NN algorithm was applied in two phases.

Although it is not possible to compare Juliusdottir et al. [38] results with others' due to incompatibilities in experimental methodologies, Juliusdottir et al. [38] point out that it is possible to make a tentative and qualified comparison.

Singh et al. [72] used all data for training. This is not the preferred method as it is not possible to get an accurate estimate of the performance of selected subsets on unseen data if all data is used for training. The preferred method is k -fold cross-validation where the algorithm is trained on some data and other (hitherto unseen) data samples can be used to get an accurate estimate of how the selected subset would perform on unseen data. Topon & Iba [58] used a 50/50 split for training and testing. Although this is better than Singh et al. method, k -fold cross-validation is preferred over 50/50 split.

The result obtained by Juliusdottir et al. [38] on prostate cancer dataset is slightly better than Topon & Iba [58] but not as good as Singh et al. [72].

Li et al. [48] obtained 65% accuracy from a 50 gene subset using EA/ k -NN approach. Liu et al. [49] achieved 91.94% classification accuracy using Leave One out Cross Validation (LOOCV).

Juliusdottir et al. [38] obtained 82.35% classification accuracy after applying EA/ k -NN algorithm to the whole dataset. This improved to 94.12% when EA/ k -NN method was applied in two phases. This result is better than previous reported work on this dataset.

Table 2.1 below summarises results for various approaches to feature selection and classification in prostate cancer dataset.

2.5 Multi-Objective Evolutionary Algorithms

As described in Chapter 1, it was decided that the two-phase EA/ k -NN algorithm introduced by Juliusdottir et al. [38] would be used as the baseline algorithm for this thesis. As explained in Chapter 1, the objective function of the EA in the

| Algorithm | Classification Accuracy | Length |
|--------------------------------|-------------------------|--------|
| PMBGA / SVM | 94.12% | 24 |
| PMBGA / SVM | 82.35% | 6 |
| Signal / noise stats with k-NN | 93.12% | 16 |
| EA/k-NN | 87.04% | 28 |
| Two-phase EA/k-NN | 88.86% | 28 |

Table 2.1: Summary of results for various approaches to feature selection and classification in prostate cancer dataset.

two-phase EA/ k -NN algorithm combines the classification error of the model with the length of the feature subset encoded by it to create a single objective value. A variable, termed α , is used in the two-phase EA/ k -NN algorithm to control the trade-off between preference for classification accuracy and preference for shorter feature subsets. The effectiveness of the algorithm depends on the correct tuning of this parameter.

Therefore, the trade-off between the accuracy and the length of the model has to be calculated for each dataset the algorithm is applied to as this trade-off is critical in obtaining the best performance from the algorithm. The best way of estimating the optimal value for this parameter for a dataset is to carry out a series of experiments on the dataset. This is both resource-intensive and time-consuming.

One way of avoiding this resource-intensive and time-consuming step is to use a multi-objective EA in the two-phase EA/ k -NN algorithm instead of a single objective EA. The multi-objective EA maximises the accuracy of the model while simultaneously minimising the length of the selected feature subset. Therefore, with a multi-objective EA, there is no need for the parameter α .

As this problem has more than one objective (classification accuracy and the length of the feature subset) that needs to be optimised, it can be classified as a Multi-objective Optimisation Problem (MOP). A MOP can be mathematically formulated as follows [85]:

$$\begin{aligned}
&\text{minimise} && F(x) = (f_1(x), \dots, f_m(x))^T \\
&&& \text{s.t.} \quad x \in \Omega
\end{aligned} \tag{2.2}$$

Where Ω is the decision space and $x \in \Omega$ is a decision vector. $F(x)$ consists of m objective functions $f_i : \Omega \rightarrow R, i = 1, \dots, m$, where R^m is the objective space [85].

The objectives in 2.2 conflict with each other and therefore multi-objective optimisation algorithms concentrate on finding *Pareto optimal solutions* [76]. Pareto optimality was first introduced by Edgeworth and Pareto [85]. A solution is termed Pareto optimal if it is non-dominated with respect to all the objectives [76]. In this case, a solution can be termed Pareto optimal if there are no other solutions that are better either on classification accuracy or the length of the chromosome. All the Pareto optimal solutions, when plotted in objective space is termed as the Pareto front [76]. EAs are population based, therefore, they are able to approximate the whole Pareto front in a single run [85].

Konak et al. [1] states that the aims of a multi-objective optimisation approach should be:

1. The best-known Pareto front should be as close as possible to the true Pareto front. Ideally, the best-known Pareto set should be a subset of the Pareto optimal set.
2. Solutions in the best-known Pareto set should be uniformly distributed and diverse over the whole Pareto front in order to provide the decision-maker a true picture of trade-offs.
3. The best-known Pareto front should capture the whole spectrum of the Pareto front. This requires investigating solutions at the extreme ends of the objective function space.

These aims are readily applicable to the feature selection and classification problem in large datasets as the ultimate aim is to identify feature subsets that can classify unseen samples with accuracy and robustness. If a multi-objective EA can fulfil the above aims, then the resulting Pareto front should contain feature subsets that give domain experts (e.g. oncologists) a good indication as to the promising areas for further research or putative drug targets.

Tables 2.2, 2.3 & 2.4, adapted from Konak et al. [1], lists the following multi-objective GAs:

- Vector Evaluated Genetic Algorithm (VEGA)
- Multi-Objective Genetic Algorithm (MOGA)
- Weight-Based Genetic Algorithm (WBGA)

- Niched Pareto Genetic Algorithm (NPGA)
- Random Weighted Genetic Algorithm (RWGA)
- Pareto Envelope-based Selection Algorithm (PESA)
- Pareto Archived Evolution Strategy (PAES)
- Nondominated Sorting Genetic Algorithm (NSGA)
- Nondominated Sorting Genetic Algorithm-II (NSGA-II)
- Strength Pareto Evolutionary Algorithm (SPEA)
- Strength Pareto Evolutionary Algorithm-2 (SPEA-2)
- Rank Density-based Genetic Algorithm (RDGA)
- Dynamic Multi-Objective Evolutionary Algorithm (DMOEA)

2.6 Related Results from Literature for the Datasets Used in This Thesis

2.6.1 Leukaemia Dataset

Bangpeng and Shao [4] introduced a novel Additive Non-parametric Margin Maximum for Case-Based Reasoning (ANMM4CBR) method for feature selection and classification in DNA microarray datasets. They managed to obtain a best classification accuracy of $97\% \pm 2.3$ with 50 features on the leukaemia dataset. With 10 features, they managed to obtain an accuracy of $96.3\% \pm 2.4$.

Zhu et al. [86] used a Memetic feature selection method with Filter Ranking (FR), Approximate Markov Blanket (AMB) & Affinity Propagation (AP) for feature selection and classification in the leukaemia dataset. They managed to obtain 98.08% accuracy with 28.1 features.

Debnath and Kurita [15] used an evolutionary approach together with an SVM. Their approach selects new subsets of features based on the estimates of generalisation error of the SVM and frequency of occurrence of the features in the evolutionary approach. With this method, they managed to obtain an accuracy of 100% with 3 features on the leukaemia dataset.

| Algorithm | Fitness assignment | Diversity mechanism | Elitism | External population | Advantages | Disadvantages |
|------------------|--|--|--------------|---------------------|---|--|
| VEGA [70] | Each sub population is evaluated with regards to a different objective | No | No | No | First MOGA Straightforward implementation | Tend to converge to the extreme of each objective |
| MOGA [19] | Pareto ranking | Fitness sharing by niching | No | No | Simple extension of single objective GA | Usually slow convergence Problems related to niche size parameter |
| WBGA [26] | Weighted average of normalised objectives | Niching Predefined weights | No | No | Simple extension of single objective GA | Difficulties in non convex objective function space |
| NPGA [31] | No fitness assignment, tournament selection | Niche count as tie-breaker in tournament selection | No | No | Very simple selection process with tournament selection | Problems related to niche size parameter Extra parameter for tournament selection |
| RWGA [55] | Weighted average of normalised objectives | Randomly assigned weights | Yes | Yes | Efficient and easy to implement | Difficulties in non convex objective function space |
| PESA [11] | No fitness assignment | Cell-based density | Pure elitist | Yes | Easy to implement Computationally efficient | Performance depends on cell sizes Prior information needed about objective space |

Table 2.2: Some well known multi-objective Genetic Algorithms adapted from Konak et al. [1] - part I.

| Algorithm | Fitness assignment | Diversity mechanism | Elitism | External population | Advantages | Disadvantages |
|---------------------|---|--|---------|---------------------|--|--|
| PAES [40] | Pareto dominance is used to replace a parent if offspring dominates | Cell-based density as tie-breaker between offspring and parent | Yes | Yes | Random mutation hill-climbing strategy Easy to implement Computationally efficient | Not a population based approach Performance depends on cell sizes |
| NSGA [75] | Ranking based on non-domination sorting | Fitness sharing by niching | No | No | Fast convergence | Problems related to niche size parameter |
| NSGA-II [14] | Ranking based on non-domination sorting | Crowding distance | Yes | No | Single parameter (N) Well tested Efficient | Crowding distance works in objective space only |
| SPEA [88] | Raking based on the external archive of non-dominated solutions | Clustering to truncate external population | Yes | Yes | Well tested No parameter for clustering | Complex clustering algorithm |
| SPEA-2 [87] | Strength of dominators | Density based on the k-th nearest neighbor | Yes | Yes | Improved SPEA Make sure extreme points are preserved | Computationally expensive fitness and density calculation |

Table 2.3: Some well known multi-objective Genetic Algorithms adapted from Konak et al. [1] - part II.

| Algorithm | Fitness assignment | Diversity mechanism | Elitism | External population | Advantages | Disadvantages |
|-------------------|--|-------------------------------------|------------------|---------------------|--|---|
| RDGA [51] | The problem reduced to bi-objective problem with solution rank and density as objectives | Forbidden region cell-based density | Yes | Yes | Dynamic cell update Robust with respect to the number of objectives | More difficult to implement than others |
| DMOEA [82] | Cell-based ranking | Adaptive cell-based density | Yes (implicitly) | No | Includes efficient techniques to update cell densities Adaptive approaches to set GA parameters | More difficult to implement than others |

Table 2.4: Some well known multi-objective Genetic Algorithms adapted from Konak et al. [1] - part III.

2.6.2 Prostate Cancer Dataset

Mundra and Rajapakse [54] introduced an improved version of Support Vector Machine - Recursive Feature Elimination (SVM-RFE) by incorporating a Minimum-Redundancy Maximum-Relevancy (MRMR) filter. With this approach, they managed to isolate 10 features with $98.29\% \pm 2.30$ accuracy from the prostate cancer dataset.

2.6.3 Ovarian Cancer Dataset

Zhu et al. [86] also used ovarian cancer dataset with their Memetic feature selection approach. They managed to obtain 99.52% accuracy with 9 features.

2.6.4 Colon Cancer Dataset

Guyon et al. [25] proposed a feature selection method that used SVM based on Recursive Feature Elimination (RFE). They demonstrated that this method is capable of selecting a subset of 4 genes that yielded 98% classification accuracy on the colon cancer dataset using Leave One Out Cross Validation (LOOCV).

Peng et al. [60] combined a GA with SVM for feature selection and classification in the colon cancer dataset. With 12 features, using LOOCV, they managed to obtain 93.55% accuracy.

2.6.5 Breast Cancer Dataset

Bolón-Canedo et al. [6] reported 68% classification accuracy on the breast cancer dataset with 10 features using SVM-RFE algorithm. Sardana, Agrawal & Baljeet [69] used an Incremental Formulation of Trace of Ratio of Scatter Matrices (IFTRSM) and minimum Redundancy and Maximum Relevance (mRMR) filter for feature selection. Linear Discriminant Classifier (LDC) and k -NN were used for classification. Their results are shown in Table 2.5.

| Classifier | LOOCV | | 10-fold | | 5-fold | |
|------------|-----------------|-----------|-----------|------------------|----------|------------------|
| | IFTRSM | mRMR | IFTRSM | mRMR | IFTRSM | mRMR |
| k -NN | 86.6(30) | 83.51(31) | 63.48(24) | 69.07(27) | 61.2(25) | 66.78(27) |
| LDC | 100(19) | 77.32(22) | 64.31(6) | 68.16(47) | 64.19(6) | 65.61(47) |

Table 2.5: Breast cancer dataset results by Sardana et al. [69]. The best performing classification method for each cross-validation technique is highlighted in bold face.

As shown in Table 2.5, LOOCV achieved the best results. Classification accuracy gets progressively lower as the number of folds in cross-validation is decreased.

Chapter 3

Study of Parameters and Multi-Objective Approach in Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm

As explained in Chapter 2, Juliusdottir et al. [38] showed that the two-phase EA/ k -NN algorithm is capable of achieving comparable, if not better, results for feature selection and classification in two large DNA microarray datasets. Furthermore, they pointed out that the two-phase EA/ k -NN algorithm has many advantages over other FS techniques.

Although Juliusdottir et al. [38] achieved very good results with the two-phase EA/ k -NN algorithm, it may be possible to achieve even better results by tuning the parameters used in the algorithm so that it performs well across a number of datasets.

The two parameters optimised by the two-phase EA/ k -NN algorithm is the classification accuracy of the selected feature subset and the length of the selected feature subset. The algorithm maximises the classification accuracy while minimising the length of the selected feature subset. Juliusdottir et al. [38] combined these two objectives into a single objective using the Equation 2.1. This equation uses a parameter labelled α that controls the trade-off between classification accuracy

and the length of the selected subset of features. Therefore, it is necessary to do preliminary experiments in order to determine a good value for α that gives the optimal performance for the two-phase EA/*k*-NN algorithm. As each dataset has unique characteristics, α may be different from dataset to dataset.

Therefore, a logical extension to the two-phase EA/*k*-NN is to replace the single objective EA with a multi-objective EA. Multi-objective EA can optimise both the classification accuracy and the length of the selected feature subset simultaneously without the aid of any extra parameters.

This chapter presents a comprehensive study of the parameters used in the two-phase EA/*k*-NN algorithm as well as an investigation into replacing the single objective EA with a multi-objective EA in the two-phase EA/*k*-NN algorithm.

3.1 Algorithms

Two main algorithms were used in all experiments:

1. A modified version of the Two-Phase EA/*k*-NN algorithm introduced by Juliusdottir et al. [38]
2. A Two-Phase Multi-Objective Evolutionary Algorithm (MOEA)

Both algorithms were written in Java. 3-fold cross-validation was used in all experiments.

3.1.1 Two-Phase Evolutionary Algorithm/*k*-Nearest Neighbours

This is a single objective, generational, elitist EA that uses *k*-NN as the classifier / objective function.

Pseudo code for the EA/*k*-NN Algorithm:

1. Create a random population P & calculate the initial values for the objective function for all the chromosomes in P
2. Repeat for $NUM_GENERATIONS$
 - Select $(P - elite_count)$ pairs of parents using rank based selection

- Apply crossover operator to each pair of parents to produce a child
- Apply mutation operator to children
- Calculate the values for the objective function for each child
- Delete non-elitist individuals from P
- Copy children into vacant spots in P

Chromosomes in the EA are integer encoded and variable in length. Genes in the chromosomes range from 1 to n where n is the number of features in a sample in the dataset.

Please refer to Figure 2.5 for an illustration on how an integer encoded chromosome can be used to filter a subset of features from the original dataset.

Goldberg and Deb [21] showed that an optimised implementation of roulette wheel selection can run in $O(n^2)$ time and rank based selection can run in $O(n \ln n)$ time. This means that rank based selection should perform faster compared to roulette wheel selection. Initial work in this thesis showed that rank based selection was not only faster but was no less accurate than roulette wheel selection. Therefore, it was decided that rank based selection would be used in thesis instead of roulette wheel selection that Juliusdottir et al. [38] used in their version of the two-phase EA/*k*-NN algorithm.

During the rank-based selection step, each chromosome in the population was assigned a rank, based on the value returned by the objective function. Then, a probability was calculated for each chromosome using Equation 3.1:

$$probability(x) = \frac{rank(x)}{(n * (n + 1))/2} \quad (3.1)$$

Where:

- x = the chromosome for which the probability is being calculated
- n = total number of chromosomes in the population

This ensures that the best chromosomes in the population have a higher probability of being selected.

The value of the objective function for chromosome x can be calculated using the following equation:

$$f(x) = (1 - class_acc) + (n/\alpha) \quad (3.2)$$

Where:

- $class_acc$ = classification accuracy for the chromosome. Classification error, which should be minimised, is $(1 - class_acc)$.
- n = size of the gene subset encoded by this chromosome. This is also minimised.
- α = parameter controlling the trade-off between preference for accuracy and preference for smaller subset sizes.

The classification accuracy of a chromosome (X) can be calculated using the following procedure:

- Let T be the number of correct classifications for X .
- Set T to zero.
- For each sample in the training or test set:
 - Calculate the Euclidean distance from the current sample to all the other samples in the training/test set. Euclidean distance is calculated using genes encoded by X .
 - Classify current sample using the classes of k -NNs
 - If the k -NN classification matches the known classification of the sample, add 1 to T
- Classification accuracy of $X = (T/\text{total number of samples in the set})$

The algorithm aims to minimise the length of a chromosome while maximising the classification accuracy. These two values are combined to produce a single metric as the value of the objective function for the single objective algorithm. As it is difficult to combine a value that is minimised with a value that is being maximised, classification error $(1 - class_acc)$ was used instead of classification accuracy. This can be easily combined with the length of a chromosome to produce a single metric.

Other parameters for the algorithm included:

- Initial chromosome size = 30 (determined after running a series of experiments using chromosomes with different initial sizes, ranging from 10 to 400)

- Population size = 50 (determined after running a series of experiments using different population sizes, ranging from 10 to 100)
- Number of Neighbours (k in k -NN) = 3 (Taken from Juliusdottir et al. [38])
- Number of generations = 2000 (determined after running a series of experiments using different numbers of generations, ranging from 1000 to 10000)
- $\alpha = 1000$ (determined after running a series of experiments using different values for α , ranging from 0.001 to 100000)

Feature selection in DNA microarray data involves selecting a subset of informative genes from a large set of genes. When the classification performance of the entire subset of informative genes is compared to the classification performance of a single gene from the subset, the single gene may not perform as well as the subset of genes. This is due to the fact that, in cases where DNA microarray data is associated with cancers, it is common to find that more than one gene is associated with the underlying cancer [38]. Therefore, the classification performance depends on the composition of the selected subset of genes.

Falkenauer [17] identified problems characterized by objective functions that depend on the composition of the selected subset of features (group of features) as *grouping problems*. Therefore, feature selection in the context of DNA microarrays can also be classified as a grouping problem. Falkenauer [17] showed that, for a grouping problem, a group-based crossover operator performed better than traditional crossover operators (e.g. single point crossover). Therefore, a modified version of the group-based crossover operator introduced by Falkenauer [17] was used in this thesis.

The following illustrates the crossover operator that was used in this thesis (genes shown within square brackets are for illustration only):

- Let Parent 1 (P1) be [3,4,5,29,38]
- Let Parent 2 (P2) be [4,29,38,61,120,122]
- Let I be genes common to both [4,29,38]
- Let J be leftover genes after taking common genes out of both parents [3,5,61,120,122]
- Let X be random number of genes from J [5,61,122]

- Let child be union of I & X [4,29,38,5,61,122]

The mutation operator added, removed or modified (with 33.3% probability) a gene in each child so that the child would only contain unique genes.

During Phase 1, EA/*k*-NN algorithm was run on all the features (i.e. no dimensionality reduction). Then, all the unique genes present in the final populations were pooled to create a much smaller dataset. During Phase 2, EA/*k*-NN was run on this reduced dataset.

3.1.2 Multi-Objective Two-Phase Evolutionary Algorithm/*k*-Nearest Neighbours

The aim of the multi-objective version of the algorithm is to minimise both the classification error ($1 - \text{class_acc}$) and the length of a chromosome simultaneously. The ultimate objective is to isolate a set of Pareto-optimal solutions that form a Pareto front.

Zitzler et al. [44] showed that there are specific features associated with test functions that can cause multi-objective evolutionary algorithms problems in converging to the Pareto-optimal front. They showed that elitism is an important factor in overcoming these obstacles by showing that SPEA outperformed all the other tested algorithms including NSGA. They have indicated that when elitism is introduced into NSGA, it obtains similar performance to SPEA. This confirms that elitism plays a major role in the performance of multi-objective EAs. Therefore, it was decided that elitism should be used in the multi-objective version of the EA/*k*-NN algorithm implemented in this thesis. In this respect, the multi-objective version of the algorithm implemented in this thesis is similar to NSGA-II. The algorithm implemented here also brings together features from MOGA and PAES. Therefore, it can be classified as a hybrid of these algorithms. For example, the algorithm implemented here has a feature similar to the Pareto ranking of MOGA. It is also similar to PAES in that it keeps an archive of the Pareto optimal solutions during the execution of the algorithm. Then, that archive is output as the Pareto front at the end of the run.

Pseudo code for the Multi-objective EA/*k*-NN Algorithm:

1. Create a random population P and evaluate initial classification error of each chromosome in P

2. Initialise the archive
3. Calculate domination count for each individual in P and add non-dominated individuals to the archive
4. Repeat for $NUM_GENERATIONS$
 - Select $(P - elite_count)$ pairs of parents
 - Apply crossover operator to each pair of parents to produce children
 - Apply mutation operator to children
 - Evaluate the classification error of each child
 - Calculate domination count for each child and add non-dominated children to the archive
 - Re-calculate the domination count in the archive & remove dominated individuals from the archive
 - Keep $elite_count$ non-dominated (or as close to non-dominated as possible) individuals in P and delete the rest
 - Copy children into vacant spots in P

A chromosome dominates another chromosome if it is better on both objectives. Figure 3.1 illustrates this concept. The domination count for a chromosome (X) is the number of chromosomes that dominate X . A chromosome with a domination count of 0 is called a non-dominated chromosome.

Parents are selected using Binary Tournament Selection (BTS) based on domination count. Two individuals are selected randomly from the population. Out of these, the one with a domination count of 0 is selected to be the parent. If none of the individuals have a domination count of 0, then the one closest to 0 is picked. If both individuals have the same domination count, then one is picked randomly out of the two. This procedure is repeated again to select the second parent.

During Phase 1, the multi-objective version of the algorithm was run on the complete set of features (i.e. no dimensionality reduction). Then, unique genes present in the Pareto front (archive) were pooled together to form a reduced dataset that was used for Phase 2.

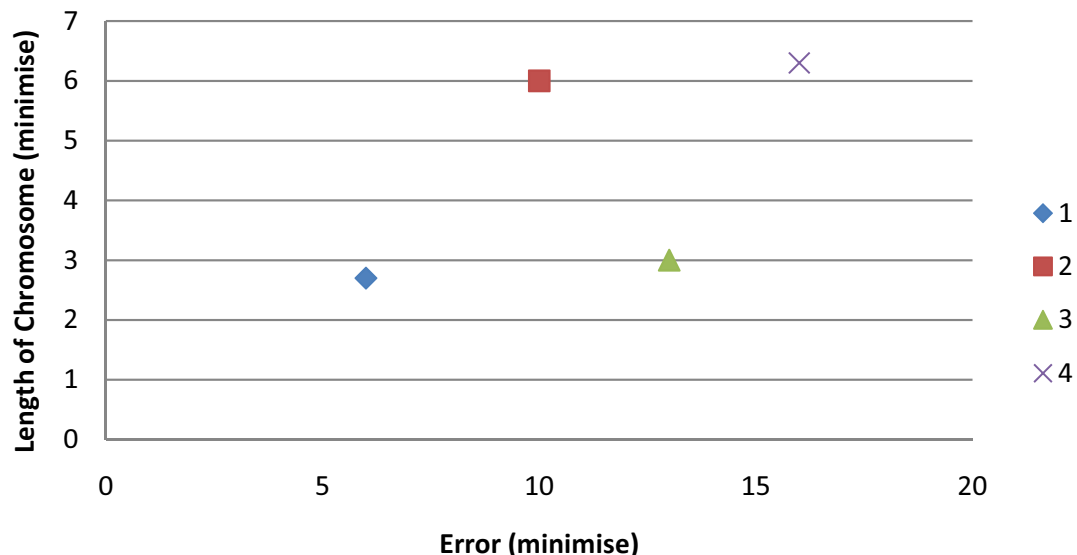


Figure 3.1: The concept of dominated solutions. Solution 1 is non-dominated as solutions 2, 3 & 4 are worse than solution 1 on both objectives. Solutions 2 & 3 do not dominate each other as solution 2 is better than 3 on accuracy but solution 3 is better on length. Both 2 & 3 are dominated by solution 1. Solution 4 is dominated by solutions 1, 2 & 3.

3.2 Datasets

The following datasets were used for all the following experiments (please refer to 1.3 for more information on these datasets):

- Leukaemia Dataset (DNA Microarray Data). This dataset was divided into 3 folds. Each fold contained 1/3 of randomly picked ALL samples and 1/3 of randomly picked AML samples.
- Ovarian Cancer Dataset (SELDI-TOF Proteomics Data). This dataset was also divided into 3 folds with an equal distribution of cancer and non-cancer samples across the folds.
- Prostate Cancer Dataset (DNA Microarray Data). This was also divided into 3 folds with an equal distribution of cancer and non-cancer samples across the folds.

Although five datasets are used for Chapters 4 and 5 below, the aim of this Chapter is to investigate the best way to arrange the two-phase EA/*k*-NN algorithm with regards to parameters, then, compare the results with Juliusdottir et al. [38]. Juliusdottir et al. [38] used two datasets for their experiments. Therefore,

it was decided that three out of five datasets, including one of the datasets used by Juliusdottir et al. [38], would be sufficient for the purpose of this Chapter.

3.3 Experiments

3.3.1 Leukaemia Dataset

3.3.1.1 Single Objective Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours

The value of the objective function is calculated by combining the classification error and the length of a chromosome into a single metric as shown in Equation 3.2.

| Error | Length | α | Objective function(error + (n/α)) |
|-------|--------|----------|---|
| 0.2 | 20 | 1000 | 0.22 |
| 0.2 | 20 | 1 | 20.2 |
| 0.2 | 20 | 0.001 | 20000.2 |

Table 3.1: Effect of α on the objective function

As shown in Table 3.1, a high value for α makes the algorithm concentrate more on reducing the error while a low value for α makes it concentrate more on reducing the length. When α is not used at all, the algorithm concentrates solely on the error. Preliminary experimentation showed that the algorithm produces much larger chromosomes when α is not used.

The aim of Two-Phase EA/ k -NN is to get rid of noise from the dataset during phase 1 and then select relevant genes during phase 2. For the algorithm to perform as expected, it is important that informative genes be kept in the gene pool. The set of experiments shown in Table 3.2 was designed to identify the best way to run the two phases and also the overall algorithm.

Abbreviations used in Table 3.2:

- SO - Single objective
- P1WO - Phase 1 without α
- P1W - Phase 1 with α
- P2WO - Phase 2 without α

| Name | Phase I | Phase II |
|--------------|------------------|------------------|
| SO P1WO | Without α | N/A |
| SO P1W | With α | N/A |
| SO P1WO P2WO | Without α | Without α |
| SO P1WO P2W | Without α | With α |
| SO P1W P2W | With α | With α |

Table 3.2: Experiments run on the leukaemia dataset

- P2W - Phase 2 with α

Explanations of the experiments listed in Table 3.2:

- SO P1WO - The algorithm concentrates on accuracy only. This means that as many informative genes as possible are selected. It was hypothesised that this would create a comprehensive gene pool for phase 2 and this would lead to better overall results.
- SO P1W - The algorithm concentrates both on accuracy and length. In this mode, the algorithm selects fewer genes compared to SO P1WO. Running the algorithm with and without α for phase 1 enables the determination of the best way to run phase 1 of the two-phase EA/ k -NN.
- SO P1WO P2WO - The algorithm is run without the effect of α during both phases. This enables testing to ascertain whether high accuracy can be obtained at the expense of length.
- SO P1WO P2W - The algorithm selects as many genes as possible during phase 1. During phase 2, the algorithm selects as few genes as possible with high accuracy. It was hypothesised that, in this mode, the algorithm would produce short chromosomes with high accuracy. This is because the algorithm is able to concentrate more on finding small subsets with high accuracy during phase 2 as it deals with a dataset with a reduced level of noise.
- SO P1W P2W - The algorithm selects as few genes as possible during phase 1. It then selects even fewer genes during phase 2. It was hypothesised that this should produce extremely small chromosomes with good accuracy.

As the dataset is divided into three folds, 3 runs of the algorithm were carried out in the following manner for each experiment

- Train on fold 0 & 1 of the dataset. Pick the chromosome with the best accuracy and the chromosome with the best length for validation. In certain cases, one particular chromosome may have the best length as well as the best accuracy. Validate these two chromosomes using fold 2 of the dataset (unseen data as far as these chromosomes are concerned).
- Repeat the same procedure using folds 0 & 2 as the training set and fold 1 as the validation set.
- Repeat the same procedure using folds 1 & 2 as the training set and fold 0 as the validation set.

3.3.1.2 Multi-Objective Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours

This algorithm minimises both objectives (classification error and length of chromosome) simultaneously. As the overall output of single objective EA/ k -NN depends on choosing an effective value for α , it was assumed that multi-objective EA/ k -NN would perform better as there is no need to pick an effective value for α . It was also assumed that running multi-objective EA/ k -NN in two phases may give better results. The following experiments were designed to test this.

- MO P1 (Multi-Objective Phase 1) - Run the algorithm on the whole dataset. Isolate unique genes from Pareto front at the end.
- MO P2 (Multi-Objective Phase 2) - Run the algorithm on unique genes from Phase 1.

3.3.2 Ovarian Cancer & Prostate Cancer Datasets

These two datasets were tested in exactly the same way as the leukaemia dataset.

3.4 Results & Discussion

3.4.1 Initial Parameter Tuning: Chromosome Size

A set of experiments was carried out on the ovarian cancer dataset in order to determine the effect of the initial size of a chromosome on the performance of the

algorithm. The results from this set of experiments are shown in Figure 3.2.



Figure 3.2: The effect of initial chromosome size on the value of the objective function. The value of the objective function of the initial randomly populated chromosomes decreases as the size of the chromosome is increased. This corresponds to an increase in the classification performance of these chromosomes. The value of the objective function during training stays low initially but increases as the size of the chromosome is increased corresponding to a degrading classification accuracy. The value of the objective function during testing follows the same trend as training albeit with markedly worse values due to over-fitting.

The average value of the objective function for the 50 initial random chromosomes over 10 repeated runs decreased as the size of the chromosome was increased. This decrease is due to increasing classification accuracy as the objective function is calculated by looking at classification error ($1 - \text{class_acc}$). This indicates that there is an optimal subset of features that is significant in classification performance. A chromosome has a better chance of getting a classification correct as the number of informative genes that it encodes increases. The chromosomes were populated with randomly selected genes to start with. Therefore, the more random genes that you encode in a chromosome, the better chance that particular chromosome has in having informative genes and therefore obtaining better classification accuracy.

However, the initial rate of decrease in the value of the objective function decreases as the size of the chromosome is increased. This is due to two reasons:

1. As the length is increased, the effect of α means that the value of the objective function increases. This counteracts the effect of classification performance of the selected subset of genes. Therefore, the rate of decrease in the value of the

objective function decreases as the size of the chromosome is increased.

2. As the length is increased, the chromosome encodes more and more noisy genes. This leads directly to degraded classification performance and therefore leads to an increase in value of the objective function which in turn slows down the rate of decrease of the value of the objective function.

In contrast, after 1000 generations while training, the value of the objective function generally stays very low indicating very good classification performance. At very small initial chromosome sizes (e.g. 10), the training value of the objective function is slightly worse than medium sized chromosomes (e.g. 30 - 50). This is due to the fact that the algorithm, with only 10 features, has not had enough time to cover a significant portion of the landscape in order to select the optimal subset of features. On the other hand, medium sized chromosomes with increased number of features give the algorithm a better chance of discovering genes associated with the optimal subset. As the size of the chromosome is increased, the performance starts degrading due to the algorithm not having had sufficient time to filter out the noise encoded in the chromosomes.

The value of the objective function while testing on unseen data follows a similar pattern to the training performance. However, test values are markedly worse than training values. This is to be expected as there is a certain amount of over-fitting that happens during the training process. Over-fitting helps keep the training values of the objective function very low. However, on unseen data, over-fitting generally degrades the classification performance.

In order to get a better understanding of the effect of the length of the chromosome on classification accuracy, the initial set of experiments was extended to include three-fold cross-validation. Figure 3.3 shows the average validation values for the objective function of 50 chromosomes after 1000 generations versus the initial size of the chromosomes. There is a clear correlation between the size of the initial chromosomes and the classification performance. Mid-sized chromosomes (20 - 50) obtain the best classification performance on unseen data.

3.4.2 Initial Parameter Tuning: The Effect of α

Juliusdottir et al. [38] used Equation 2.1 for calculating the value of the objective function for a chromosome. An analysis of this equation for the effect that α had on

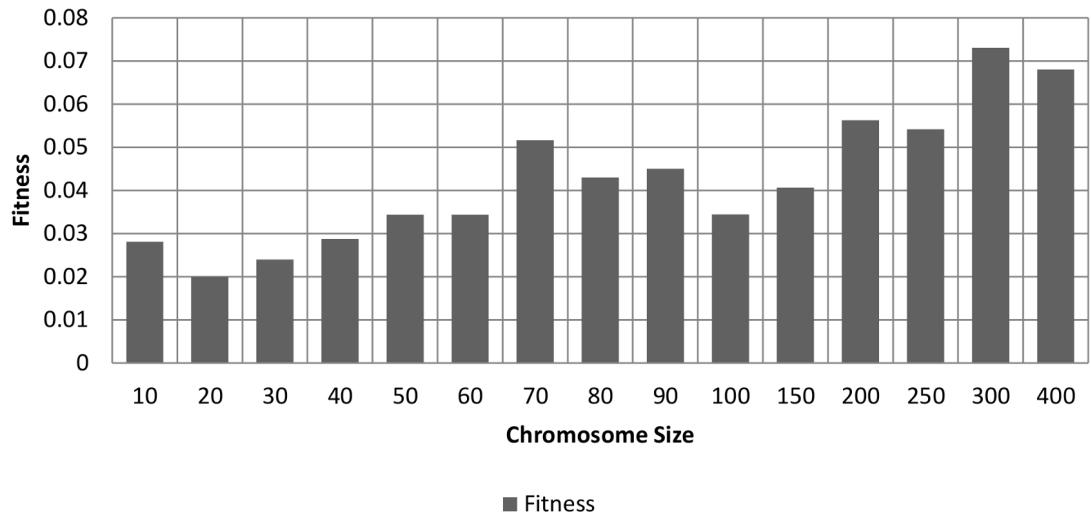


Figure 3.3: The effect of initial chromosome size on 3-fold cross-validated values of the objective function. Average cross-validated objective function values of chromosomes versus the initial size of the chromosome.

the length of the chromosome on the ovarian cancer dataset is shown in Figure 3.4.

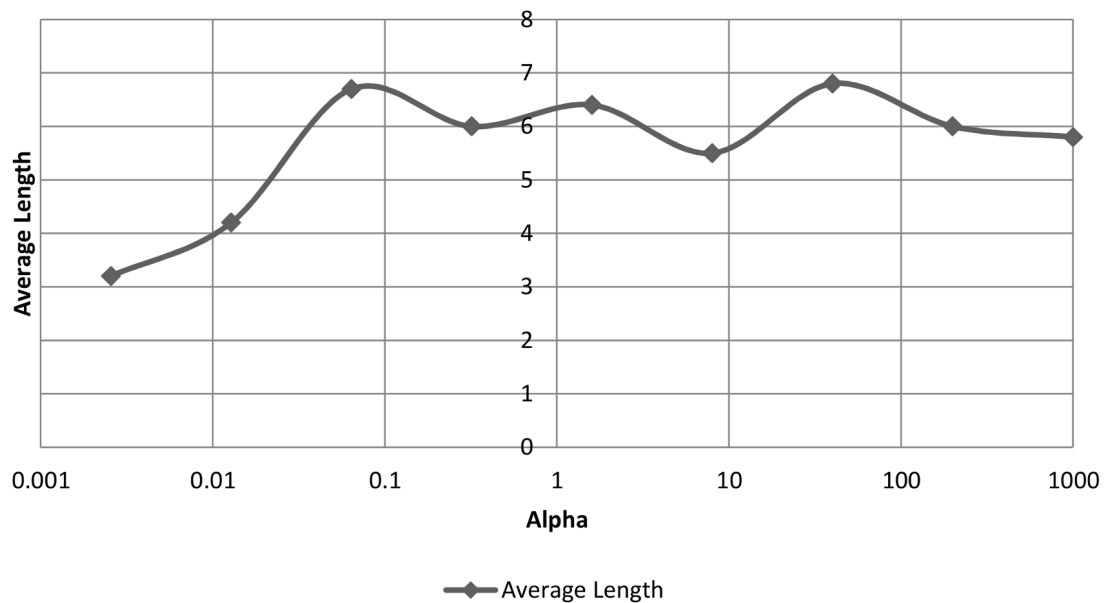


Figure 3.4: The effect of α on the final length of chromosomes after 2000 generations.

As can be seen from Figure 3.4, the relationship between average length of the chromosome and α is weak. A stronger relationship between the final length of the chromosome and the value of α is desirable as this enables experiments to be run with better control. Therefore, research was carried out into obtaining a stronger relationship between α and the length of the chromosome which resulted in the Equation 3.2.

With Equation 3.2, in theory, if α is 1 and length of chromosome is 30, then the graph of the resulting objective function values is shown in Figure 3.5

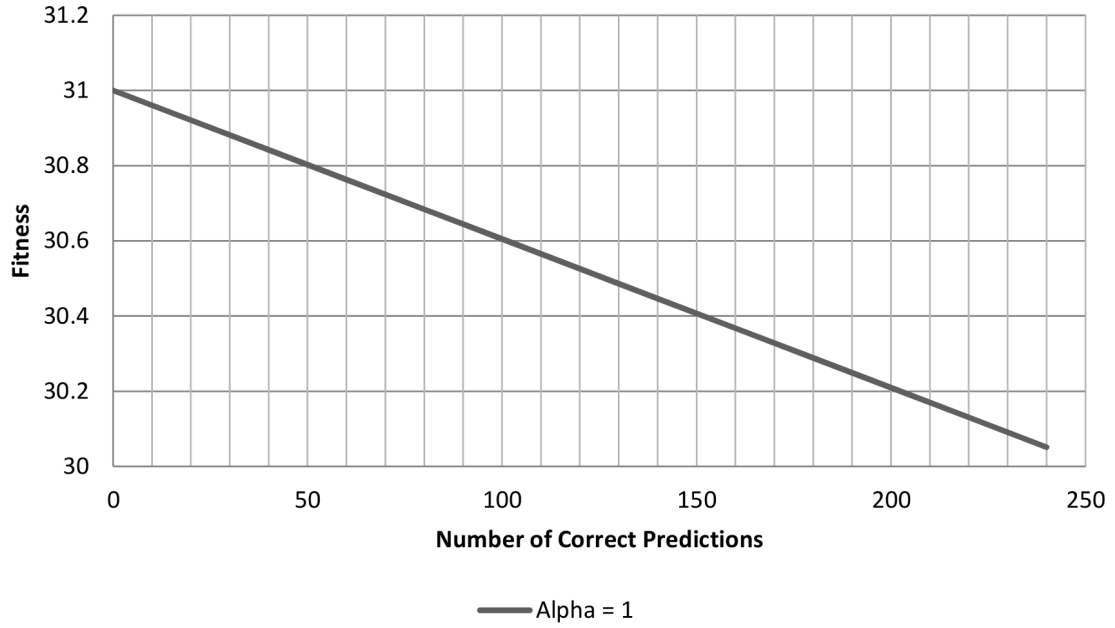


Figure 3.5: Theoretical values produced by Equation 3.2 when the length of the chromosome is 30 and α is 1.

As accuracy increases (increasing number of correct classifications), the value of the objective function decreases. Therefore, the EA has to minimize the value of the objective function. Figure 3.6 shows the resulting objective function values when α is changed from 10 to 1. As expected, the gradient of the line is the same. However, the magnitude of the value changes with α .

This equation was then tested on the ovarian cancer dataset and the results are shown in Figure 3.7. As expected, the average length of the chromosome increases as α is increased. In Figure 3.7, the α value 1000000 is used as a place-holder for illustrating the length of the chromosome when α is completely omitted from the objective function.

In order to gain a better understanding of the effect that α has on the length, the chromosome with the minimum length and the chromosome with the maximum length were plotted together with the average length of 50 chromosomes after 2000 generations on the ovarian cancer dataset (Figure 3.8). This shows a strong relationship between α and the length of the final chromosomes. It shows that with the new equation, experiments can be run with much better control of the final length of the chromosomes.

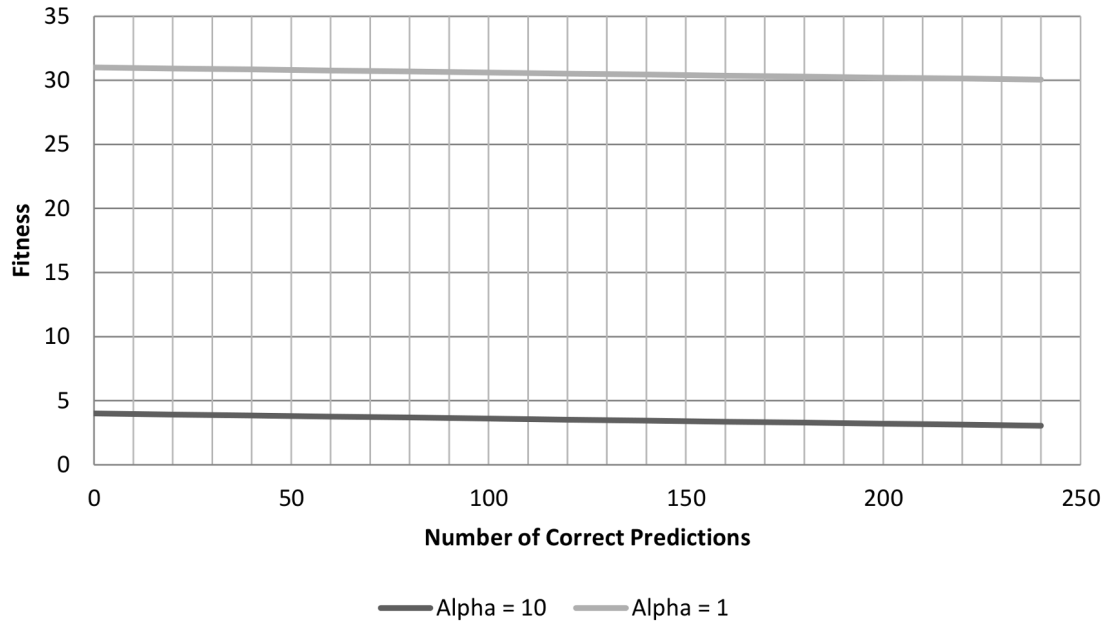


Figure 3.6: Theoretical values produced by Equation 3.2 when the length of the chromosome is 30 and α is 10, and when α is 1.

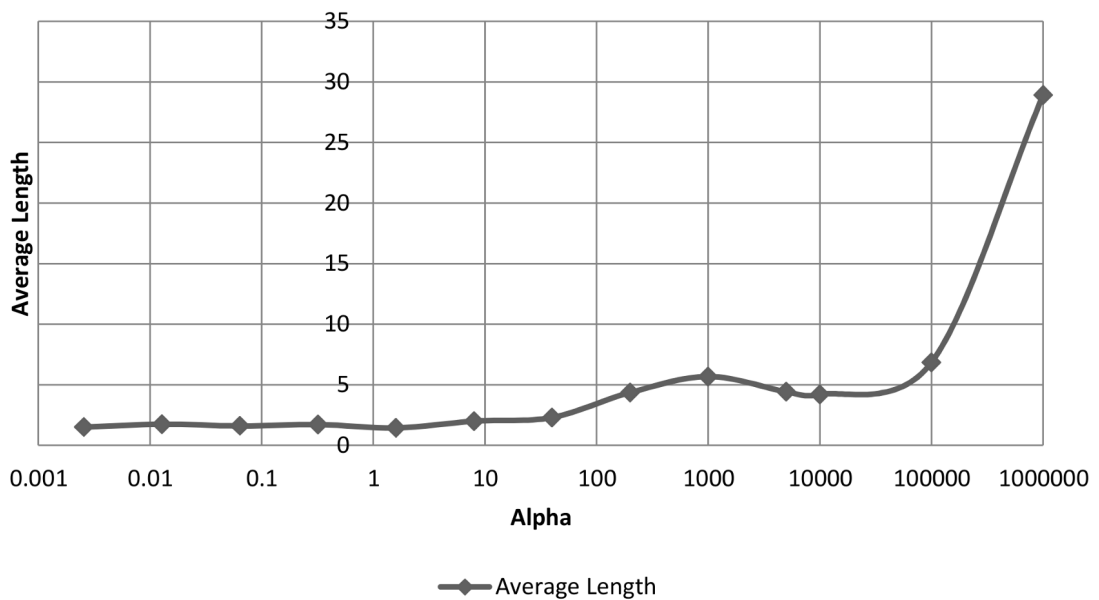


Figure 3.7: The effect that α has on final chromosomes on the ovarian cancer dataset after 2000 generations.

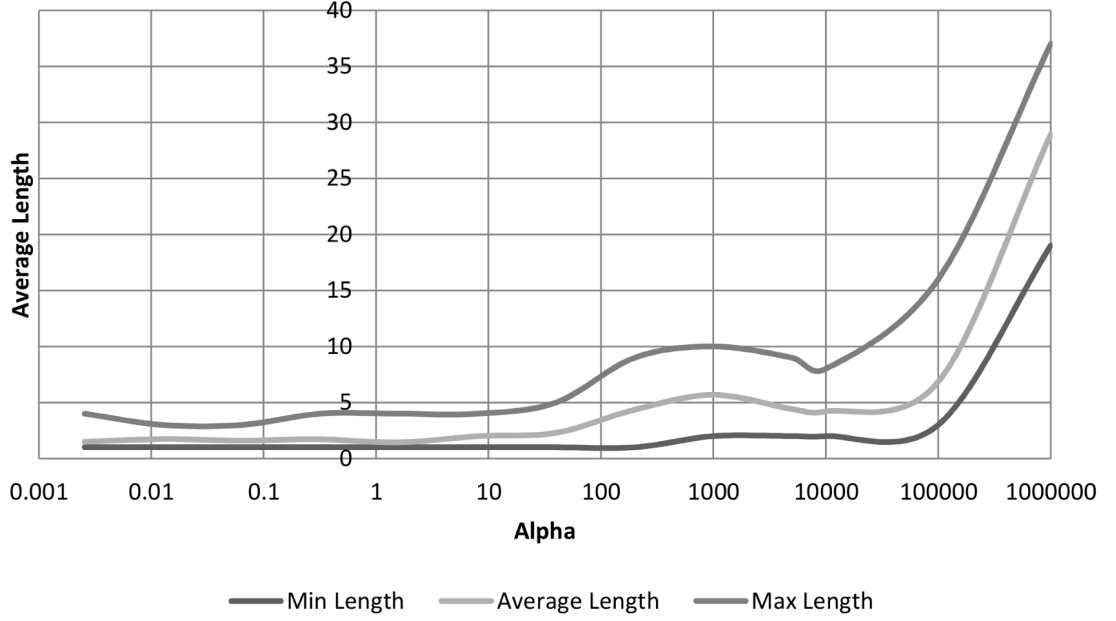


Figure 3.8: The effect that α has on the minimum, maximum and average length of final chromosomes on the ovarian cancer dataset after 2000 generations.

The EA/*k*-NN algorithm can be thought of as running in $O(i * c * (s - 1))$ time where i is the number of features encoded in a chromosome, c is the number of chromosomes and s is the number of samples in the dataset. This is due to the fact that the most time-consuming step in the algorithm is the classification step that calculates Euclidean distances for *k*-NN algorithm. In order to ascertain the classification accuracy of c , a sample (s_i) needs to be classified by looking at the Euclidean distance from s_i to all the other samples in the dataset ($s - 1$ samples). Then, *k*-NNs for s_i are selected by looking at the distances and s_i is classified according to the majority of its neighbours. If the predicted classification of s_i is accurate, then that is counted as one correct classification and the same procedure is repeated for sample $s_i + 1$, then, for $s_i + 2$, and so on until all the samples of the dataset have been classified. The classification accuracy of c , then, is the total number of correct classifications divided by the number of total classifications.

The two-phase EA/*k*-NN algorithm can be thought of as running in $O((i * c * (s - 1)) * 11)$ time. This is due to the fact that phase one of the algorithm is repeated 10 times in order to collect features that form the reduced dataset for phase two. The time taken to run both the two-phase EA/*k*-NN and EA/*k*-NN algorithms increases linearly with an increase in either the size of the dataset or the number of chromosomes used in the algorithm.

In other words, the EA/*k*-NN algorithm can be thought of as running in $O(n)$ time whereas the two-phase EA/*k*-NN algorithm can be thought of as running in $O(n * 11)$ time.

3.4.2.1 Effect of α on the Leukaemia Dataset

Figures 3.9 and 3.10 show the effect that α has on the length and the classification accuracy of final chromosomes for the leukaemia dataset. From these graphs, it can be concluded that a value of 1000 for α is likely to return best results for the leukaemia dataset.

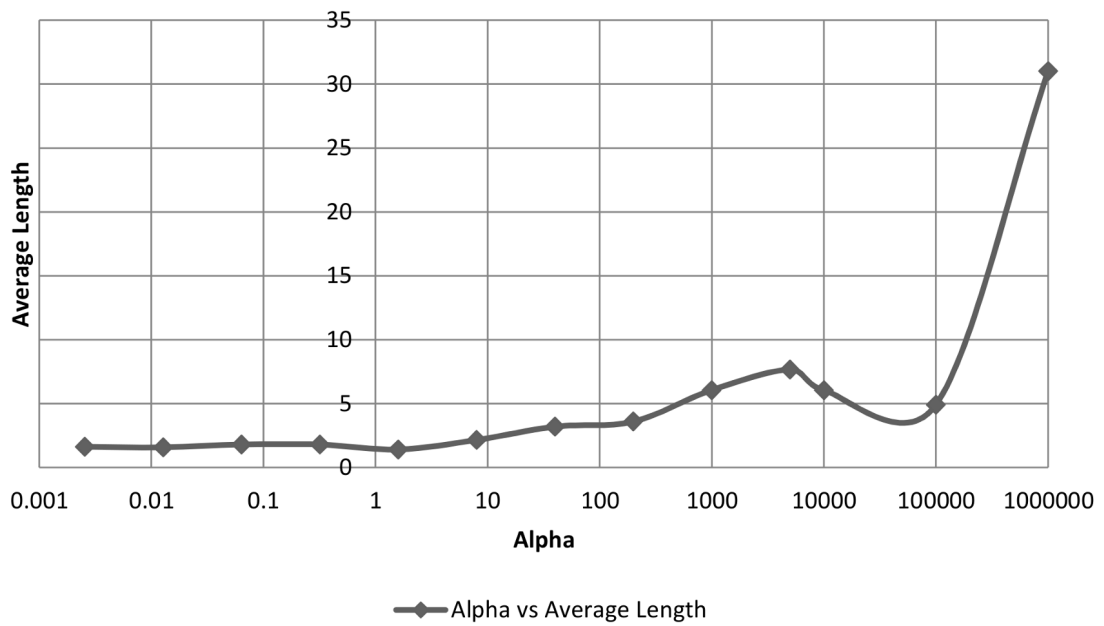


Figure 3.9: The effect that α has on the average length of chromosomes on the leukaemia dataset after 2000 generations.

3.4.2.2 Effect of α on the Prostate Cancer Dataset

Figures 3.11 and 3.12 show the effect that α has on the length and the classification accuracy of final chromosomes for the prostate cancer dataset. As with the leukaemia dataset, a value of 1000 for α looks likely to return best results for this dataset.

Overall, the best fit for α across all three datasets is 1000.

3.4.3 Leukaemia Dataset

Figure 3.13 shows the best validated chromosomes from each experiment on the leukaemia dataset. Out of the validated chromosomes, only the best chromosomes

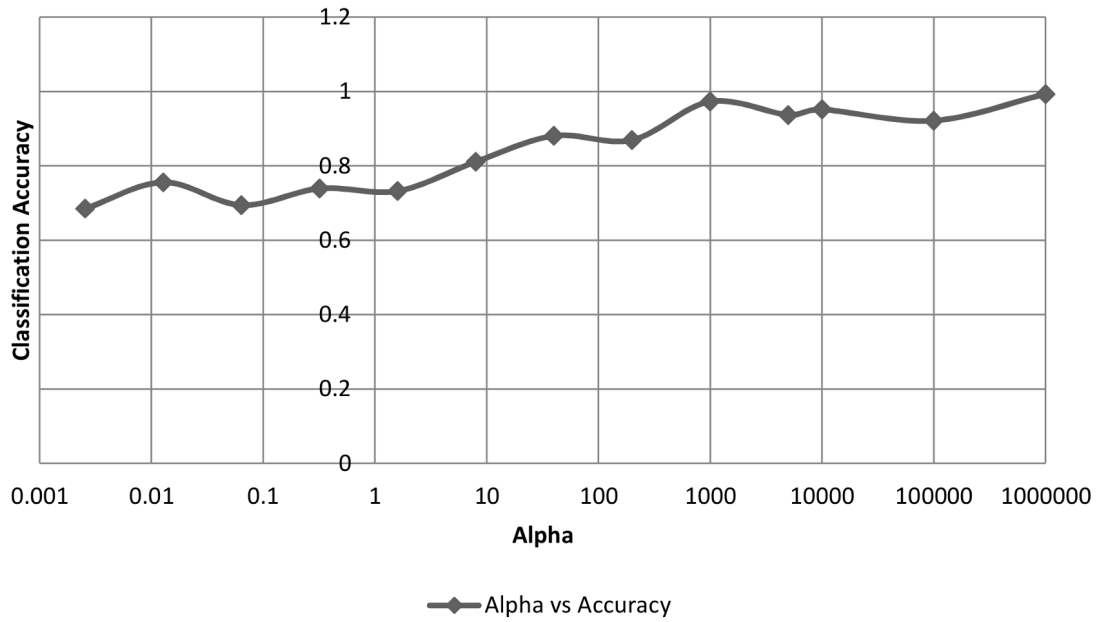


Figure 3.10: The effect that α has on the classification accuracy on the leukaemia dataset after 2000 generations.

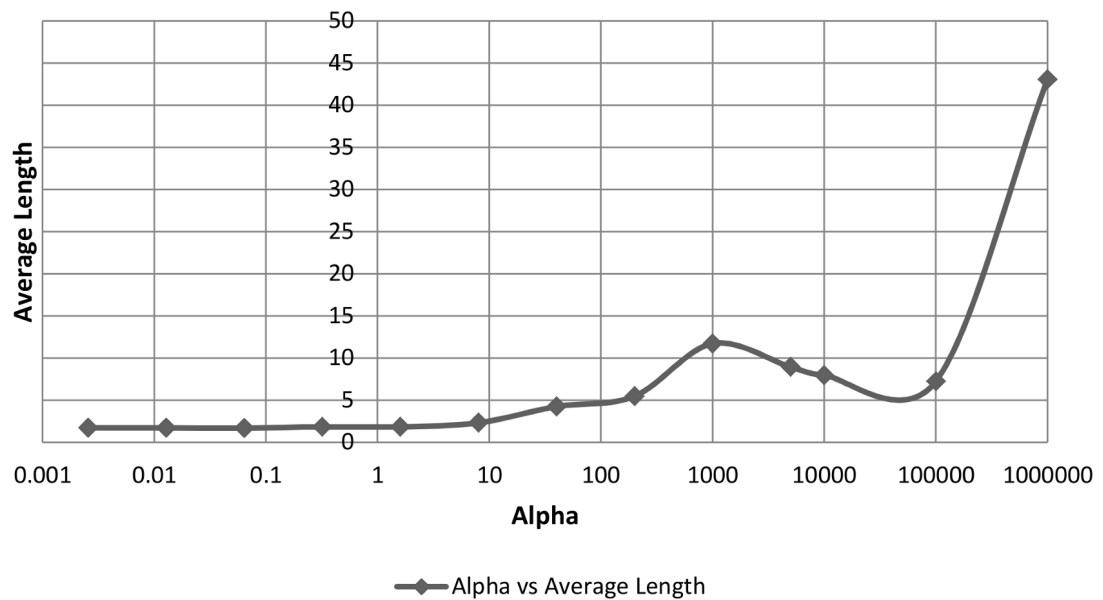


Figure 3.11: The effect that α has on the average length of chromosomes on the prostate cancer dataset after 2000 generations.

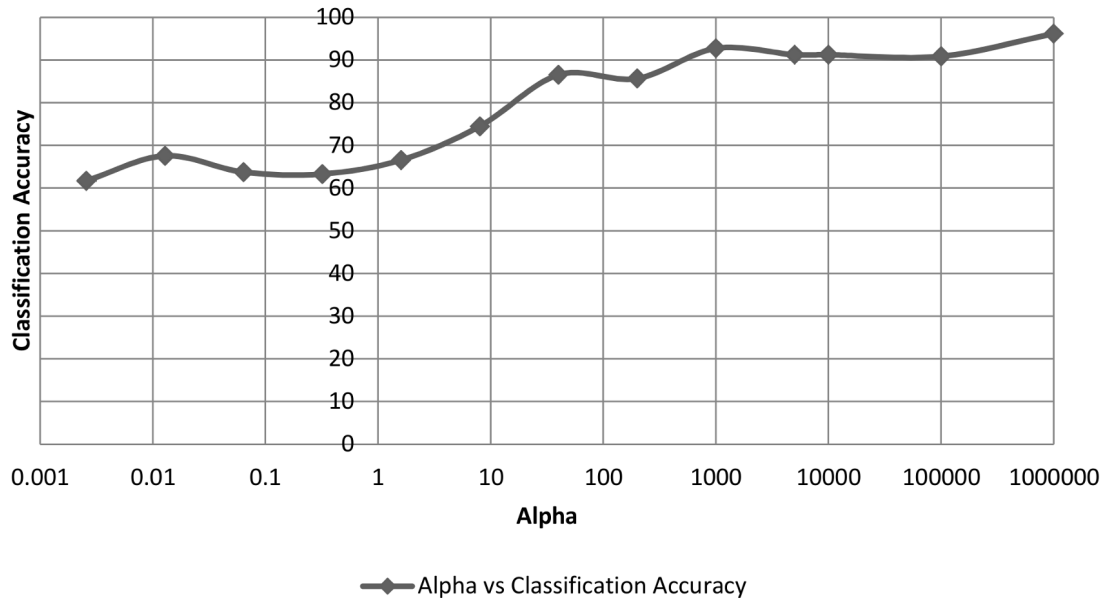


Figure 3.12: The effect that α has on the classification accuracy on the prostate cancer dataset after 2000 generations.

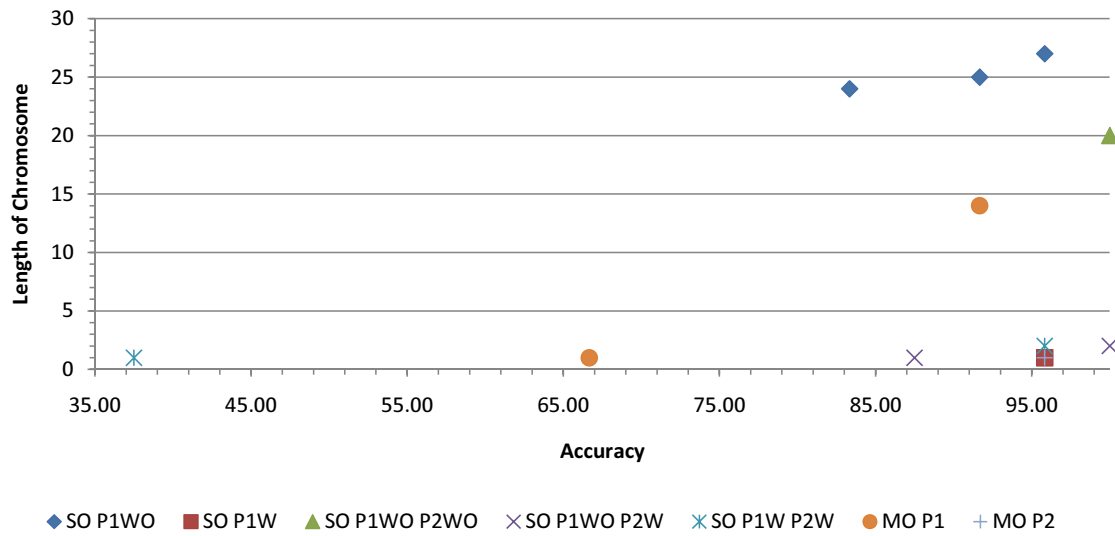


Figure 3.13: Best validated solutions for each experiment on the leukaemia dataset

for each experiment are shown.

Single objective 2 phase EA/*k*-NN has achieved 100% classification accuracy with 2 features when phase I did not take the length of the chromosome into account and phase II did. Two-phase multi-objective approach managed to obtain a classification accuracy of 95.83% with 1 feature.

This compares favourably with the results reported by Zhu et al. [86]. They obtained 98.08% accuracy with 28.1 features. Debnath and Kurita [15] reported 100% accuracy with 3 features. The single objective two-phase algorithm implemented here also obtained 100% accuracy but with only 2 features.

3.4.4 Ovarian Cancer Dataset

Figure 3.14 shows the best validated chromosomes for each experiment carried out on the ovarian cancer dataset.

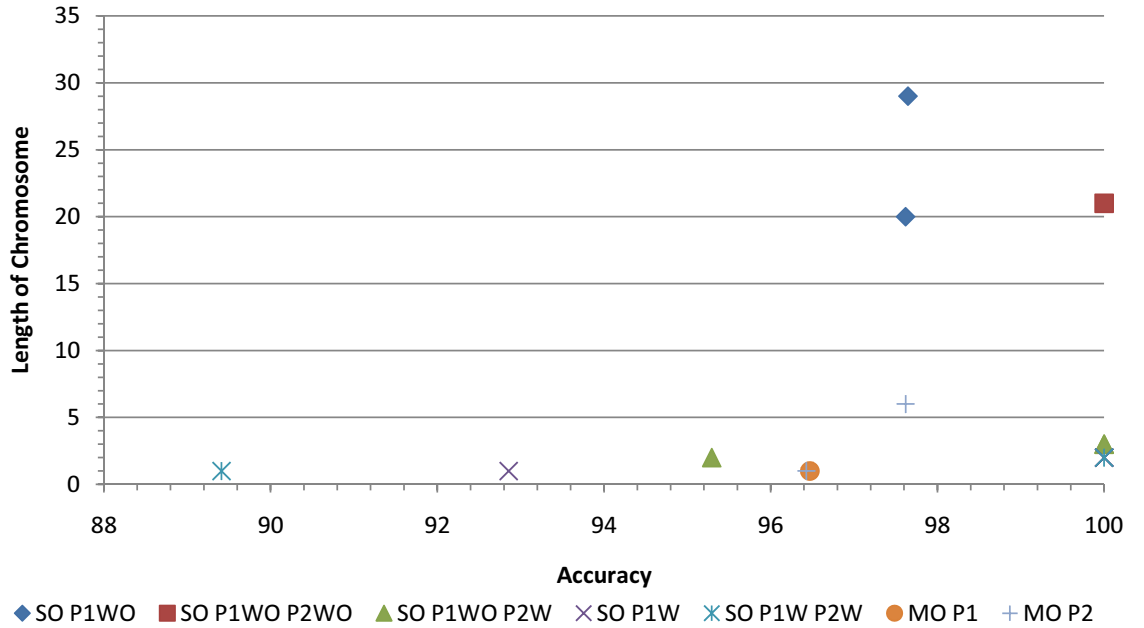


Figure 3.14: Best validated solutions for each experiment on the ovarian cancer dataset

Single objective EA/*k*-NN managed to obtain a classification accuracy of 100% with 2 features in the following cases:

- Single phase with α (taking the length of the chromosome into account)
- Two phase: phase 1 with α and phase 2 with α (both phases take length of chromosome into account)

- Two phase: phase 1 without α and phase 2 with α (phase 1 does not take length of chromosome into account while phase 2 does)

Multi-objective EA/*k*-NN managed to obtain 96.47% classification accuracy with 1 feature in a single phase. In two phases, it achieved 97.62% classification accuracy with 6 features.

On this dataset, Zhu et al. [86] reported 99.52% accuracy with 9 features. In comparison, single objective EA/*k*-NN approach presented here obtained 100% accuracy with 2 features.

3.4.5 Prostate Cancer Dataset

Figure 3.15 shows the best validated solutions from each experiment on the prostate cancer dataset.

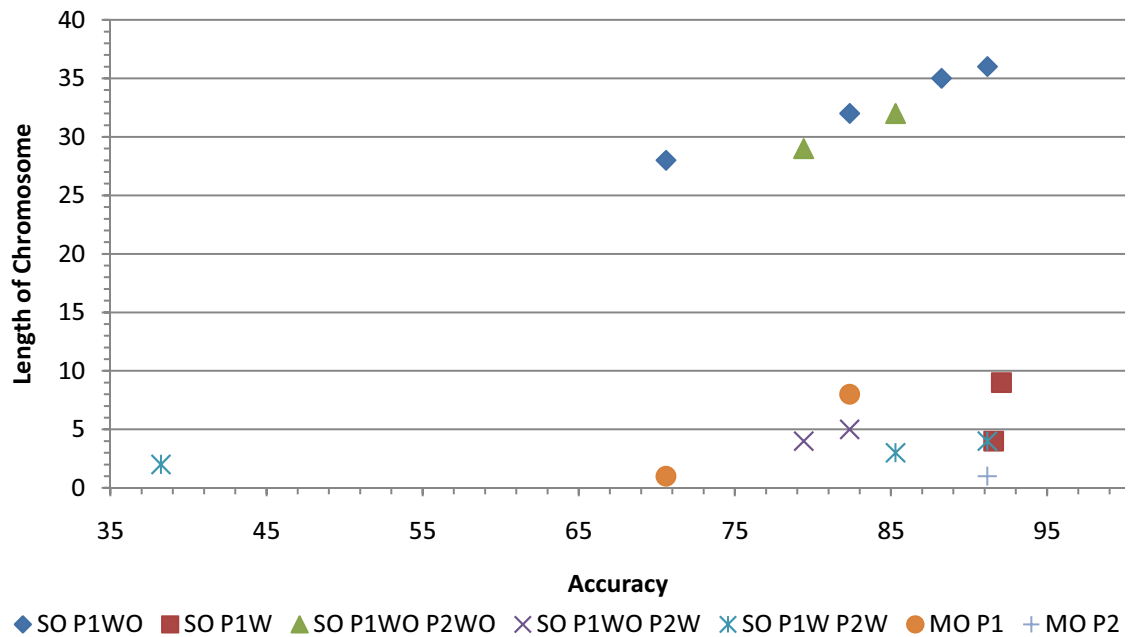


Figure 3.15: Best validated solutions for each experiment on the prostate cancer dataset

Single objective EA/*k*-NN in single phase mode (taking length into consideration) managed to obtain a classification accuracy of 92.08% with 9 features. Multi-objective EA/*k*-NN in two phases managed to obtain a classification accuracy of 91.18% with a single feature.

Mundra and Rajapakse [54] reported $98.29\% \pm 2.30$ accuracy with 10 features. Although the approach presented here managed a lower accuracy compared to Mundra

and Rajapakse, the multi-objective approach managed it (91.18% accuracy) with a single feature.

3.5 Conclusion

Results for the three datasets studied here have been reported in several papers. It is difficult to do a careful comparison since different researchers tend to use varying ways to partition the datasets into training, test and validation sets, and different researchers tend to use their own variants (very common), updates, subsets, and/or older versions of the datasets.

Also, consideration of predictive accuracy is not a main aspect of the current work, but it can nevertheless be reported that the results on unseen data that were reported in 3.4 are consistent with the general body of literature on these datasets, particularly considering work that aims to find minimal and understandable models (e.g. small subsets of genes with high predictive accuracy).

The EA/ k -NN method explored here is clearly competitive in the region of accuracy with small numbers of features. Single objective EA/ k -NN in two phases (both phases taking the length of the chromosome into consideration) has achieved the best results on all three datasets. However, it should be noted that for the two-phase EA/ k -NN to achieve these results, a considerable amount of preliminary work went into parameter tuning.

However, the multi-objective EA/ k -NN in two phases has achieved very competitive results on all three datasets (leukaemia - 95.83% with 1 feature, ovarian - 97.62% with 6 features and prostate - 91.18% with 1 feature) with no preliminary work. It is this ability to produce very competitive results without the need for preliminary work that makes the multi-objective approach look very promising for feature selection and classification in bioscience/medical datasets.

Although the multi-objective two-phase EA/ k -NN method looks very promising for feature selection and classification, particular features (genes) selected during repeated runs of the algorithm need to be compared with results obtained from other approaches. If a consistent pattern emerges, then these features can be further studied by research biologists. If different features emerge from different approaches but on a consistent basis, then further work can be carried out into why different methods select different features. In turn, this may lead to further understanding

of the underlying data.

Future work should concentrate on improving the performance of two-phase multi-objective EA/ k -NN. A map of genes most frequently selected over repeated runs of the algorithm can be created. These genes can then be compared with genes selected by other approaches in literature in order to determine how meaningful these features are and, in turn, how meaningful these methods are.

Chapter 4

An Adaptive Weights Scheme for k -Nearest Neighbours in Evolutionary Algorithm/ k -Nearest Neighbours Algorithm for Feature Selection

4.1 Introduction

As described in Chapter 3, the two-phase EA/ k -NN method proposed by Juliusdottir et al. [38] looks promising for feature selection and classification in predictive data mining. Therefore, research was carried out into tuning the parameters used in the two-phase EA/ k -NN algorithm so that it performs well across a range of datasets. An investigation was also carried out into using a multi-objective EA instead of a single-objective EA in the EA/ k -NN algorithm.

Although the results from this research were promising, there are many areas where the methodology could be improved in order to obtain even better performance from the algorithm. Therefore, the first part of this chapter looks into the areas that can be improved. In the second half of the chapter, an investigation is carried out into weighting strategies for k -NN in order to ascertain whether the performance of the algorithm can be improved by applying feature weights during the classification step.

4.2 Improvements to the Methodology

4.2.1 Cross-validation

A version of k -fold cross-validation [9] was used in Chapter 3 for validating the results obtained from all the algorithms. Usually, in k -fold cross-validation, the dataset is randomly split into k mutually exclusive subsets. The algorithm is then trained on $k - 1$ folds and tested on the remaining fold.

In Chapter 3, stratified 3-fold cross-validation was employed [41]. In stratified cross-validation, each fold contains approximately the same proportion of classes (e.g. cancer and normal) as the original dataset. The algorithm was then trained on two folds and tested on the remaining fold. This procedure was repeated two more times so that the algorithm was tested exactly one time on each fold. The reported accuracy was the average accuracy over the three cross-validation runs.

The cross-validation method employed in Chapter 3 has a few drawbacks. First of all, each experiment was repeated multiple times on each dataset. For each repeat, the split of the dataset into three folds was kept the same. An improvement to this approach would be to randomly split the dataset into three folds each time an experiment is repeated. This is warranted by the distribution of informative genes and noise within the samples. The algorithm is able to extract useful information more easily from some samples than others. This is especially the case in biological datasets where non-disease related factors may also influence gene expression profiles [63].

In some instances, if the training fold happens to contain a larger proportion of easier-to-classify samples, then the EA converges quickly. However, the performance of the selected model on unseen data may not be robust. Therefore, if the experiment is repeated 10 times, then the quality of the final result may depend on the way the samples were allocated to each fold. As described by Raser and O'Shea [63], gene expression data is stochastic by nature. There is no convenient way of determining which samples are easy to classify and which samples are harder. Therefore, the aim of feature selection should be to build robust models that are equally capable of classifying easy and hard samples. In theory, this process should be more efficient and robust if the dataset is randomly split into k -folds before each repeat of the algorithm. This is especially the case in two-phase EA/ k -NN algorithm where phase

one is repeated many times and genes selected during phase one are fed into phase two of the algorithm.

Ron Kohavi [41] points out that in truly difficult problems, the models created are most stable when leave-one-out cross-validation is employed. In case of k -fold cross-validation, the stability of the selected model increases with higher numbers for k . For example, more stable models can be built with 20-fold cross-validation compared to 10-fold cross-validation.

Although, in theory, a higher number of k should produce better results, the k that is actually used should be determined not only by trying to increase the metric, but also by looking at the nature of the dataset and the underlying problem. For example, by selecting inappropriate numbers for k , it is possible that a random partition of a dataset into k folds may in fact lead to either a training or testing fold that only contains one class of samples. If this happens to be a training fold, then the models that are built will not be suitable predictors for the underlying case.

Therefore, for the second part of this chapter, it was decided that 3-fold cross-validation would be applied to all the experiments. However, in order to make the results robust, it was decided that each repeat of the 3-fold cross-validated experiment would be carried out on a fresh random split of the dataset. With this approach, the chances of a large proportion of easily-classified samples being in the same fold across all the repeated runs are much reduced. At the same time, the algorithm gains all the benefits of 3-fold cross-validation (e.g. a large and diverse enough set of samples for training).

4.2.2 Model Selection

In traditional science, a model is statistically tested using null hypothesis testing [35, 3] in order to determine the robustness. There are a wide variety of statistical methods in the literature that concentrate on null hypothesis testing. However, in some branches of science (e.g. ecology), null hypothesis testing is being replaced with the practice of “model selection” [35]. In model selection, several competing models are tested at the same time on the same set of data. Then the results of these tests are analysed to determine the best model or make inferences about case under study.

In the context of an EA, model selection carries a slightly different set of require-

ments. One of the major requirements is the avoidance of over-fitting. This aspect of EAs was not addressed in Chapter 3. In Chapter 3, the dataset is split into 3 folds and the algorithm is trained on 2 folds. Then the best solutions are tested on the third fold (unseen data).

Models (chromosomes) are selected solely based on the training performance. This invariably leads to at least some models that have a very strong training performance but a poor performance on unseen data being selected (due to over-fitting).

This was further analysed by testing all the models generated during training on the testing fold (unseen data). If after testing all the models, the models with best testing performance are selected, then the algorithm would have produced models with almost perfect classification performance (nearly 100% classification accuracy on unseen data) across all the datasets. This indicates that it is possible to increase the general performance of the algorithm with improved model selection during training (with a view to minimising over-fitting).

However, most methods of model selection/evaluation found in the general body of literature is not suitable for use in an EA. This is due to the fact that a full implementation of these methods [3] for each model produced during each generation of the EA will amount to a great computational cost. Therefore, research needs to be carried out into simpler methods of model selection that lead to improved performance.

One simple method of achieving this is to modify the cross-validation technique. As described in section 4.2.1, 3-fold cross-validation involves splitting the dataset into three folds, training the algorithm on two out of the three folds and then validating the trained models on the remaining fold. This leads to the selection of models solely based on training performance and may lead to the selection of models that include random error or noise (over-fitting). To a certain extent, over-fitting can be avoided if the models are selected on their performance on unseen data. This can be achieved by implementing the following cross-validation technique:

- Randomly split the dataset into three stratified folds: fold A, fold B & fold C.
- Train the algorithm on fold A.
- Test the trained models on fold B.
- Rank the models by looking at their classification performance and the length.

Isolate a set of best performing models as “selected models”.

- Validate the “selected models” on the remaining fold of the dataset (fold C).
- Repeat the complete procedure two more times so that models are validated on fold A & fold B.
- Take the average classification accuracy of the best chromosomes over the three cross-validation runs as the cross validated classification performance of the algorithm.

This approach has the advantage that models are selected for validation on the basis of their performance on unseen data. Therefore, these models are less likely to be affected by over-fitting. This approach was adopted for the second part of this Chapter.

4.2.3 Setting up of Phase I & II of the Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm

In the two-phase EA/ k -NN algorithm, phase one is used for initial dimensionality reduction on large and noisy datasets while phase two is used for building robust predictive models. If phase two is to succeed in building robust predictive models, then phase one has to succeed in eliminating noise from the dataset while keeping informative genes in the gene pool so that phase two has enough genes to work with.

Also, as phase two can only work with genes selected during phase one, phase one should be given enough time to explore the fitness landscape and select promising areas. Too little time would mean that phase one will not select all the informative genes for phase two to work with, while too much time would mean that some of the informative genes will be dropped as the algorithm converges on promising areas.

The search of the fitness landscape in this context can be expressed both in terms of the number of generations and the number and the length of the initial chromosomes. Setting the number of generations to a small value would mean a premature end to the phase one search for informative genes. For the same number of generations, an EA with a higher number of larger initial chromosomes would cover more of the fitness landscape compared to an EA with a lower number of smaller initial chromosomes.

Therefore, the success of the two-phase EA/ k -NN algorithm depends, to a large extent, on the set-up of the two phases. Chapter 3 explored the different ways of setting up the two-phase EA/ k -NN algorithm (please refer to Table 3.2). As there was no firm conclusion as to which set-up was better across datasets, the first part of this chapter further explores this area. It also explores whether Rank Based Selection (RBS) or BTS is better suited for feature selection and classification in predictive data mining using the two-phase EA/ k -NN algorithm.

4.3 Adaptive Weights for k -Nearest Neighbours

k -local Hyperplane Distance Nearest Neighbour (HKNN) algorithm has been shown to perform very well in some applications [56, 57, 78]. First, prototypes for each class are selected using the k -NN algorithm. A local hyperplane is then constructed for each class using the prototypes. A sample can then be classified by looking at the distance between the sample and all of the local hyperplanes constructed during the previous step [81].

HKNN has been shown to be effective only for small values of k [78]. As the local hyperplane for each class is constructed using k prototypes for each class, the accuracy of the whole approach depends on the selection of k prototypes.

HKNN uses the k -NN algorithm for selecting prototypes. The k -NN algorithm suffers from bias in high dimensions [81]. Yang and Kecman [81] showed that these problems can be overcome by using the Adaptive Nearest Neighbour (ANN) algorithm instead of classical k -NN.

The ANN algorithm considers feature weights when selecting k -NNs of a query. Feature weights are estimated by using the ratio of between-group to within-group sums of squares [81].

Yang and Kecman [81] used ANN together with local hyperplanes in a method they termed Adaptive Local Hyperplane (ALH) for classification. They showed that ALH is capable of obtaining better classification performance on publicly available datasets compared to other commonly used classification techniques including k -NN. Juliusdottir et al. [38] showed that two-phase EA/ k -NN method is capable of obtaining comparable, if not better, results for feature selection and classification. Juliusdottir et al. [38] also argued that a highly efficient classifier such as an SVM may be counterproductive to feature selection and classification. This is due to the

fact that a highly efficient classifier will obtain a good classification performance from a sub-optimal feature subset and therefore lead to a “flattening” of the fitness landscape.

The “flattening” of the fitness landscape occurs in the following way: classification accuracy is measured as a percentage. Therefore, the maximum value possible is 100%. An optimal subset of features may yield 100% accuracy both with a sophisticated classifier (e.g. SVM) and a less sophisticated classifier (e.g. k -NN). A sub-optimal feature subset may yield 90% accuracy with k -NN but the same subset may yield 98% accuracy with SVM. The EA when combined with an SVM will identify this sub-optimal feature subset as being very close to the optimal. Therefore, the gap between sub-optimal and optimal feature subsets gets reduced and this leads to a “flattening” of the fitness landscape. It can be argued that an EA combined with k -NN will lead to less “flattening” of the fitness landscape leading to the discovery of feature subsets that are more relevant to the case under study. As a classifier, ALH is closer to an SVM than k -NN. Therefore, it was decided that ALH should not be used for the purposes of this thesis.

Taking the above into account, it was decided to carry out research into integrating a modified version of the ANN algorithm with the two-phase EA/ k -NN algorithm, in order to ascertain whether ANN would enhance the performance of the two-phase EA/ k -NN algorithm.

4.3.1 Adaptive Weights for k -Nearest Neighbours Explained

ANN uses the following equation for calculating the “Weighted Euclidean Distance” between a sample (X_i) and a query (q) [81]:

$$D(X_i, q) = \sqrt{\sum_{j=1}^d w_j (x_{ij} - q_j)^2} \quad (4.1)$$

Where:

- X_i = An instance from the training set
- q = Query for which nearest neighbours are being selected
- d = Number of features in the query
- w_j = Weight for j th feature from the weights vector for class of X_i

- x_{ij} = j th feature of training sample i
- q_j = j th feature from the query

The weight for a feature, w_j , is calculated using the following equation [81]:

$$w_j = \frac{\exp(TR_j)}{\sum_{j=1}^d \exp(TR_j)} \quad (4.2)$$

Where:

- T = A positive parameter that controls the influence of R_j on w_j .
- $R_j = r_j / \max(r_j)$
- d = Number of features in a sample

r_j is calculated using the following equation [81]:

$$r_j = \frac{\sum_i \sum_c I(y_i = c)(\bar{x}_{cj} - \bar{x}_j)^2}{\sum_i \sum_c I(y_i = c)(x_{ij} - \bar{x}_{cj})^2} \quad (4.3)$$

Where:

- \bar{x}_{cj} denotes j th component of class centroid of class c
- \bar{x}_j denotes j th component of grand class centroid
- x_{ij} denotes j th feature of sample i

4.3.2 Application of Adaptive Weights

Equation 4.3 can be applied to a dataset in the following way:

- Split the dataset into separate classes
- For each class, find the class centroid
- Use class centroids to find grand class centroid
- Then, for each feature (j) for each individual (i) in a class:
 - Let X be the total of (feature j of class centroid - feature j of grand class centroid)²

- Let Y be the total of (feature j of i - feature j of class centroid)²
- $r_j = X/Y$

r_j can then be converted into w_j using Equation 4.2.

When this procedure is applied to the dataset, the result is a ratio vector for each class in the dataset. Assuming that weights are being calculated on Table 4.1, centroids for classes A & B and grand class centroid are shown in Table 4.2.

| Sample | Features | | | | | | Class |
|--------|----------|---|----|---|---|----|-------|
| S1 | 2 | 6 | 9 | 4 | 5 | 14 | A |
| S2 | 3 | 3 | 2 | 2 | 3 | 9 | B |
| S3 | 6 | 0 | 2 | 5 | 5 | 10 | A |
| S4 | 6 | 6 | 6 | 6 | 6 | 6 | B |
| S5 | 0 | 0 | 10 | 5 | 2 | 4 | B |

Table 4.1: Sample Dataset

| Centroid | Features | | | | | |
|----------------------|----------|---|------|-------|------|------|
| Class A Centroid | 4 | 3 | 5.5 | 4.5 | 5 | 12 |
| Class B Centroid | 3 | 3 | 6 | 4.33 | 3.66 | 6.33 |
| Grand Class Centroid | 3.5 | 3 | 5.75 | 4.415 | 4.33 | 9.16 |

Table 4.2: Centroids

The following equation shows how the ratio is calculated for feature 1 for class A from the sample dataset:

$$r_1 = \frac{(4 - 3.5)^2 + (4 - 3.5)^2}{(2 - 4)^2 + (6 - 4)^2} = \frac{0.25 + 0.25}{4 + 4} = 0.0625$$

The way the cross-validation is performed has implications on the way the weights are calculated. Weights calculated as explained above takes into account all the samples of a class in the dataset. However, with the variation of the 3-fold cross-validation used here, during the training phase of the algorithm, only the training fold (1 fold out of the 3) is used. If weights are calculated for the complete dataset, then, models are exposed to test and validation data to a certain extent. Therefore, weights should only be calculated for the training fold during training. Furthermore, when classification accuracy for a model is calculated, each sample in the training fold is tested by selecting its 3 closest neighbours, classifying the sample using the majority classification of the neighbours and checking the calculated classification against the known classification of the sample. As each sample is tested, the number

of attempted classifications is equal to the number of samples in the training fold. The classification accuracy can then be calculated by looking at the percentage of correct classification. In this context, it is logical to calculate a set of weights excluding the sample that is being classified currently. As the aim is to classify the current sample by assuming that it belongs to an unknown class, if this sample is included in the weights calculation, then the classification algorithm gains access to information that it will not have when it is being used for classifying genuinely unseen, unclassified samples. Therefore, for each attempted classification during training, a decision was made to calculate a fresh set of weights that excluded the sample that is being classified.

For both testing and training, the weights calculation is slightly simpler compared to the weights calculation for training. This is because a sample from either the testing or the validation fold is classified by looking at its closest neighbours from all the training samples. This step emulates the classification of genuine, unclassified samples. There is no learning or information gain in the model at this stage. Therefore, a set of weights can be calculated that includes the whole of the training fold.

4.4 Algorithms

All algorithms were written in Java and 3-fold cross-validation was used for all experiments (where appropriate).

4.4.1 Stand-alone k -Nearest Neighbours and Weighted k -Nearest Neighbour

These algorithms contained the classifier without the EA. It was necessary to create these algorithms in order to make a direct comparison between classic k -NN and W- k -NN.

Assuming there are 100 samples (S1 ... S100) in the dataset, the stand-alone k -NN algorithm measures its performance on the dataset as follows:

1. Take S1:
 - (a) Calculate Euclidean distance from S1 to S2, S1 to S3 and so on

- (b) Take k closest distances to S1
 - (c) Classify S1 using the majority of closest neighbours
 - (d) If the classification matches with the known classification for S1, add 1 to the total
2. Repeat the above procedure for S2 and all remaining samples
 3. The classification accuracy for the dataset is $((total/numberofsamples) * 100)$

The stand-alone version of the W- k -NN algorithm includes the additional step of obtaining weighted Euclidean distances and finding the closest neighbours using that instead of normal Euclidean distance.

4.4.2 Single-objective Evolutionary Algorithm/ k -Nearest Neighbours Algorithm

This is a single objective, generational, elitist EA that uses k -NN as the classifier/objective function. Chromosomes in the EA are integer encoded and variable in length. Genes in the chromosomes range from 1 to n where n is the number of features in the dataset.

Figure 2.5 illustrates how this approach is used to filter a subset of features from the original dataset. The dataset shown in Figure 2.5 has three samples (1 normal and 2 cancer) and each sample has 6 features (genes). The chromosome encodes three features: 2, 3 & 5. Integer encoded chromosomes can be used either to create a much smaller dataset as shown in Figure 2.5 or to classify a sample. Please refer to 2.4.1 for further details on this algorithm.

4.4.3 Single-objective Evolutionary Algorithm/Weighted- k -Nearest Neighbours Algorithm

The single-objective EA/W- k -NN algorithm is identical to the single-objective EA/ k -NN algorithm apart from the fact that it uses W- k -NN for classification rather than k -NN.

In order to incorporate weights into the algorithm, the following steps were added:

- For each class, create class centroid
- Use class centroids to create grand class centroid
- For each feature in a class, use Equation 4.3 to calculate the ratio of within-group to between-groups sums of squares
- Normalise this ratio using $R_j = r_j / \max(r_j)$ (where r_j = a ratio for a feature)
- Apply Equation 4.2 to create a weights vector for each class

Then, when calculating distance between a chromosome and its neighbours, the square of the difference between features is multiplied by the weight for that feature.

4.4.4 Single-objective Evolutionary Algorithm/Weighted Centroid Classification Algorithm

k -NN in previous versions of this algorithm worked in the following manner:

- Calculate distance from a sample (X) to all the other data samples
- Go through the distances array and find the k closest neighbours

However, there was a potential problem with this approach. For example, the closest distance may be 1.5 and there may be 3 samples with that distance. The next closest distance may be 2 and there may be 2 samples with this. The third closest distance may be 2.1 and there may be 1 sample with this. So, overall we have 6 neighbours. In this case, in the previous implementation of k -NN, only the first sample having a certain distance was considered when classifying. In this example, the first sample with a distance of 1.5 may belong to class A and the other two to class B. This leads to incorrect classification.

In real-life datasets, this may be extremely rare as gene expression profiles from DNA microarrays contain real values with many decimal places. However, it was a common occurrence during testing of the algorithms with fictional datasets. Therefore, it was decided to compare the performance of k -NN and W- k -NN with a novel weighted centroid classification technique.

Centroid classification usually entails calculating the centroid for each class in a dataset, then, getting the Euclidean distance from a sample to the centroid, and

finally, classifying the sample according to the smallest distance. The approach used here differs from this approach in that the distance for each feature from the sample in question to each of the centroids is multiplied by the adaptive feature weight for that feature.

For example, if a chromosome encodes features 2 & 5, then the weighted centroid classification would be carried out like this:

Assume there are two classes (class A and class B) in the dataset.

- Calculate the distance from sample 1 to the class A centroid for feature 2 and multiply this distance by the class A weight for feature 2
- Calculate the distance from sample 1 to the class A centroid for feature 5 and multiply this distance by the class A weight for feature 5
- Sum the distances to get the total distance from sample 1 to the class A centroid. Let this be X
- Calculate the distance from sample 1 to the class B centroid for feature 2 and multiply this distance by the class B weight for feature 2
- Calculate the distance from sample 1 to the class B centroid for feature 5 and multiply this distance by the class B weight for feature 5
- Sum the distances to get the total distance from sample 1 to the class B centroid. Let this be Y
- If $X < Y$, then classify the sample as class A. If $Y < X$, classify the sample as class B. Break ties randomly.

4.5 Preliminary Experiments

4.5.1 Speed of the Algorithm

When calculating the weights, a new set of weights had to be calculated for each sample (excluding the current sample). If the dataset contained 100 samples, then in order to classify one chromosome, 100 different sets of weights had to be calculated. This, coupled with Euclidean distances, meant that the algorithm was computationally very expensive. It had to be optimised by changing the way some of the steps

of the calculation were done. This enabled the bulk of the calculation to be done beforehand. The algorithm only had to “look up” values during run-time making it much faster.

Fitness inheritance was also tried in order to determine if the execution time of the algorithm could be reduced without compromising the accuracy. Fitness inheritance was originally suggested by Smith et al. [73]. They proposed that computationally intensive fitness calculations could be done only for a portion of the population. This is in contrast to Grefenestette and Fitzpatrick’s [24] earlier approach of partially evaluating the fitness of the entire population.

Smith et al. [73] proposed evaluating the fitness of only part of the population. The rest of the population would “inherit” the fitness of their parents. However, rather than a straightforward inheritance of the fitness value, they suggest that the fitness value of a child chromosome should be derived in a simple way from the fitness values of its parents. They put forward two ways of deriving the child fitness:

- Averaged Inheritance: Child’s fitness is equal to the arithmetic mean of the parents’ fitness values.
- Proportional Inheritance: Child’s fitness is a weighted average of the fitness of the parents based on the contribution of genes from each parent to the child.

Smith et al. [73] showed that a proportional approach to deriving the inherited fitness of a child chromosome could achieve excellent results even when less than 1% of the population had their fitnesses fully evaluated.

Other ways of achieving fitness inheritance include estimating the fitness of a chromosome not only by looking at the parents but also at the neighbouring individuals. Branke and Schmidt [8] used interpolation and regression analysis to estimate (or predict) the fitness of a child chromosome. They concluded that it is possible to either achieve a better fitness in a given time or to reach a given fitness level in reduced time without compromising the quality of the solutions.

Barbour, Corne & McCall [5] applied fitness inheritance strategies suggested by Smith et al. [73] to the cancer chemotherapy treatment schedule optimisation problem and found that averaged inheritance can produce an 80% saving on model evaluations (computationally intensive fitness evaluation step) at 95% inheritance (only 5% full fitness calculations).

As the weights scheme for k -NN and k -NN itself are both computationally expensive and fitness inheritance is a promising solution for lowering the computational cost of EAs, it was decided that fitness inheritance should be tested in order to ascertain if it can be successfully applied to the problem of feature selection and classification in large datasets.

Averaged fitness inheritance was implemented and tested for various rates of inheritance ranging from 10% to 95% inheritance in each generation. Averaged fitness inheritance was also tested for a percentage of generations. For example, if fitness was inherited for all the chromosomes but in every other generation, then this would indicate a 50% inheritance with reference to the number of generations. The results indicated that the version of the fitness inheritance implemented in this thesis was not efficient in cutting down on computational time while achieving small feature subsets with good classification performance.

Due to the fact that only one method of fitness inheritance (averaged inheritance, suggested by Smith et al. [73]) was investigated here, it can be argued that fitness inheritance remains a valid subject for future research. The nature of the problem in feature selection and classification means that some features contribute more to the fitness of an individual than others. Therefore, estimating the fitness of a child using the fitness of the parents without paying attention to the selected genes may lead to poor results. For example, the parents of a child chromosome may encode some important genes as well as some noisy genes. If the child encodes mainly the informative genes from both parents, then the fitness of the child will obviously be much higher than the arithmetic mean of the parents. Therefore, it is logical to assume that, in this case, proportional inheritance may yield better results.

4.5.2 Estimating T

T is a parameter used in the Equation (4.2) that calculates the weights for the W - k -NN algorithm. The calculated weights depend on T , therefore, the classification accuracy also depends on the value of T . A series of experiments was carried out on each dataset to determine if there are different values for T for each dataset that improve the classification performance.

4.6 Main Experiments

The main experiments were carried out in two sections:

- Further exploration of the best way to set up the two-phase EA/ k -NN algorithm prior to the application of feature weights
- Exploration of the application of feature weights to the k -NN algorithm

4.6.1 Set-up of the Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm

One of the conclusions of Chapter 3 was that it was possible to obtain much better results from the algorithm when the objective function took the length of the chromosome into account during phase two of the algorithm. This is due to the fact that the models built during phase two need to be evaluated both for their classification accuracy and the length of the feature subset encoded by the model.

The selection method employed was also tested in order to learn whether BTS or RBS would perform better in feature selection and classification across a number of datasets.

In summary, the following experiments were carried out:

- BTS
 - P1W
 - P1WO
 - P1WO P2W
 - P1W P2W
- RBS
 - P1W
 - P1WO
 - P1WO P2W
 - P1W P2W

Where:

- P1W: Phase one run with the effect of the length of the chromosome taken into account
- P1WO: Phase one run without the effect of the length of the chromosome taken into account
- P2W: Phase two run with the effect of the chromosome taken into account

4.6.2 Exploration of Feature Weights

The best method from 4.6.1 was then tested on the following methods:

- EA/W- k -NN algorithm
- EA/Weighted Centroid Classification algorithm

4.7 Results and Discussion

4.7.1 Speed of the Algorithm

One of the main problems with the algorithm was the time that it took to calculate weights as this step had to be done for each data sample and for each chromosome. This then had to be repeated for each generation.

In order to cut down on time, it was decided that all Euclidean distances would be pre-calculated and stored in a serialized Java object. This object is then read from disk to memory when needed. The same procedure was applied to the weights.

However, this turned out to be impractical as the size of the object storing distances and weights (using double precision numbers) was much larger (over 3.5GB) than the available memory in 32-bit Windows systems. “Look up” method only became viable by using single precision numbers and optimising the storage method. It was decided that the loss of precision when going from double to single precision in Java would not be an issue for this algorithm.

Table 4.3 shows a summary of results for k -NN classifier on all three datasets. These are averages over 10 runs on a PC with Intel Core 2 Duo processor at 3.00GHz with 2.00GB of RAM.

| Dataset | Calculate everything on the fly | “Look up” values |
|-------------------|---------------------------------|------------------|
| Ovarian Dataset | 17.96 days | 4.54 days |
| Prostate Dataset | 2.61 days | 0.61days |
| Leukaemia Dataset | 0.73 days | 0.17days |

Table 4.3: Calculating everything on the fly vs. look up for k -Nearest Neighbours

4.7.2 Estimating T: Stand-alone k -Nearest Neighbours vs. Weighted k -Nearest Neighbour

4.7.2.1 Stand-alone k -NN and W- k -NN on the Leukaemia Dataset

Figure 4.1 shows the accuracy of stand-alone k -NN on the leukaemia dataset for a range of values of T.

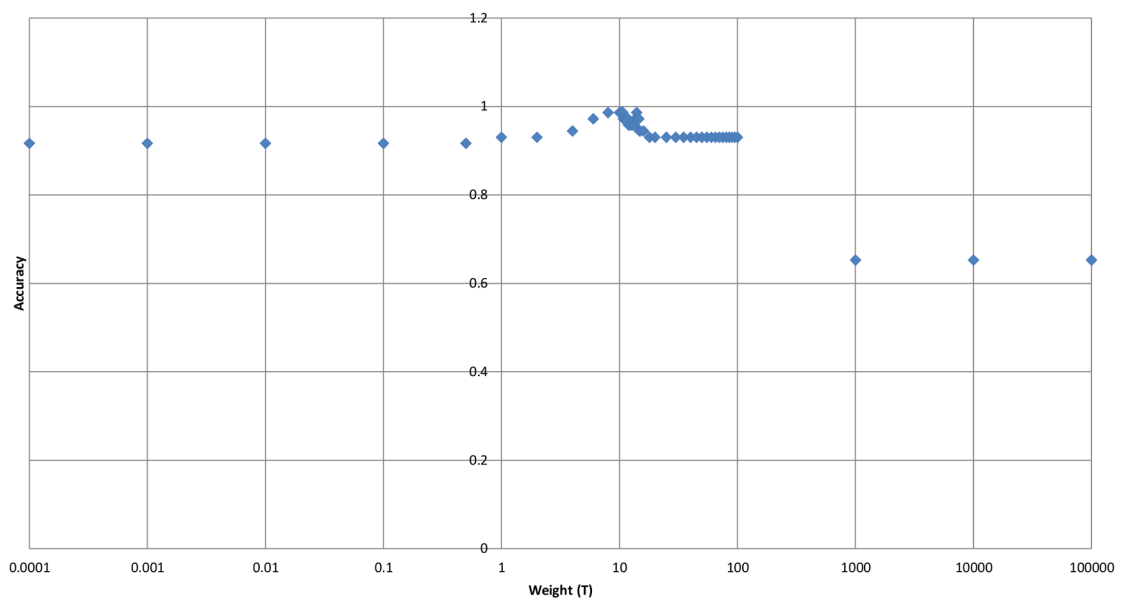


Figure 4.1: Accuracy of stand-alone k -Nearest Neighbours on the leukaemia dataset for a range of values for T.

As shown in Figure 4.1, the performance of W- k -NN classifier is clearly better than the performance of classical k -NN. The performance of classical k -NN is shown as the first point of the graph where $T = 0$. Classical k -NN achieved 91.67 % accuracy on the leukaemia dataset whereas W- k -NN managed to achieve 98.61% accuracy. This was achieved for values of T between 8 to 10.7.

4.7.2.2 Stand-alone k -Nearest Neighbours and Weighted k -Nearest Neighbour on the Prostate Cancer Dataset

Figure 4.2 shows the accuracy of stand-alone W- k -NN on the prostate cancer dataset for a range of values of T.

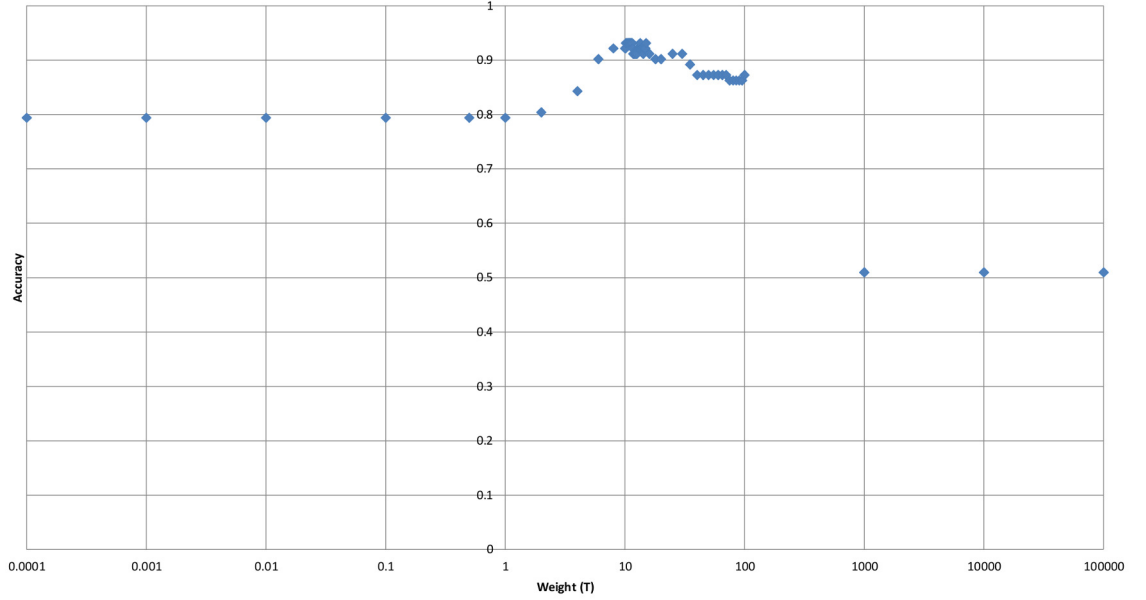


Figure 4.2: Accuracy of stand-alone Weighted k -Nearest Neighbour on the prostate cancer dataset for a range of values for T

As with the leukaemia dataset, W- k -NN achieved a much better accuracy compared to k -NN. On the prostate cancer dataset, k -NN achieved 79.41% accuracy while W- k -NN achieved 93.14% accuracy which is a considerable improvement. W- k -NN achieved this with values of T between 10.2 to 11.5.

4.7.2.3 Stand-alone k -Nearest Neighbours and Weighted k -Nearest Neighbour on the Ovarian Cancer Dataset

Figure 4.3 shows the accuracy of stand-alone W- k -NN on the ovarian dataset for a range of values of T.

On the ovarian cancer dataset, classical k -NN achieved an accuracy of 94.86% while W- k -NN achieved an accuracy of 97.23%.

Overall, W- k -NN clearly outperformed k -NN on all three datasets. However, in order to gain this performance from W- k -NN, the parameter labelled T has to be adjusted to suit each dataset. On the leukaemia dataset, the best result was obtained when T was set between 8 and 10.7. On the prostate cancer dataset, T

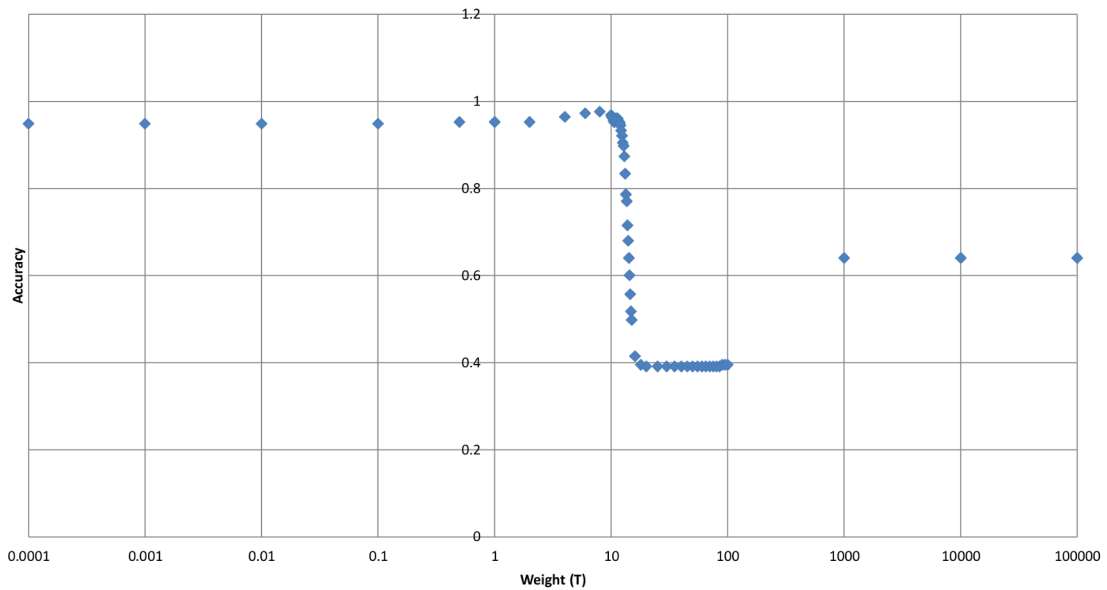


Figure 4.3: Accuracy of stand-alone Weighted k -Nearest Neighbour on the ovarian cancer dataset for a range of values for T

had to be set between 10.2 to 11.5 for the best results. However, on the ovarian dataset, T had to be set to between 6 to 8 to obtain the best results. This clearly indicates that, in order to get the best results from W - k -NN, T has to be adjusted for each dataset.

The prostate dataset is known to be a difficult dataset for classification tasks. This is due to the fact that the classification accuracy reported for this dataset in the general body of literature is lower than, for example, the ovarian cancer dataset. Therefore, it seems that there is a link between the “difficulty” of the dataset and the value of T : higher values of T need to be used to get better results from “difficult” datasets. However, more research needs to be carried out before a firm conclusion can be made.

4.7.3 Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours vs. Two-Phase Evolutionary Algorithm/Weighted- k -Nearest Neighbours

One of the main problems identified with the two-phase EA/ k -NN approach is how best to set up the two phases. Juliusdottir et al. [38] set up the two phases in the following way:

- Run Phase I ten times (400 generations with 80 chromosomes each with an initial length of 400) and collect genes present in the best chromosomes at the end.
- Run Phase II once with the genes collected from all ten runs of Phase I for 100 generations with 30 chromosomes each with an initial length of 100.

However, these settings cannot be used in general for all the datasets. In order to obtain the best results, the algorithm needs to be set up for each dataset by carrying out preliminary research into various combinations of parameters (e.g. the number of generations to run phase one for and the number of generations to run phase two for).

Model selection is also a problematic area. Juliusdottir et al. [38] pick genes for phase two by looking at the final best chromosomes from phase one. This may not be the best approach in selecting genes for phase two. The point of running the algorithm in two phases is to reduce noise present in the dataset during phase one and then to learn robust models during phase two. However, the stopping point for phase one is critical for the success of phase two. If phase one is stopped too early, then genes selected for phase two will still contain noise. If phase two is stopped too late, then a certain amount of noise creeps back into the selected pool of genes due to over-fitting. It is essential to reduce the error introduced by over-fitting, especially in datasets considered as “difficult” for classification.

In order to determine if a general end-point can be achieved across all datasets, a set of experiments was carried out as outlined below.

4.7.3.1 End-Point Experiments for Phase I

It was decided that in order to compare the performance of the two-phase EA/ k -NN algorithm with the two-phase EA/W- k -NN algorithm, an experimental procedure that generalised well across all the datasets used here had to be adopted for the two-phase EA/ k -NN algorithm. Then, the EA/W- k -NN algorithm can be run using the same procedure and a direct comparison can be made between the two algorithms.

The first parameter that had to be standardised was the number of generations that phase one of the algorithm should be run for. This was tested by splitting each dataset into two folds, training the algorithm on one fold and testing the generated models on the remaining fold. The folds were then swapped and the procedure was

repeated. The average best testing accuracy for each generation was then plotted in order to identify any patterns that may emerge.

As explained in 4.6.1, the following configurations of the two-phase EA/ k -NN algorithm were tested:

- BTS
 - P1W
 - P1WO
- RBS
 - P1W
 - P1WO

The following set of graphs illustrate the results for the leukaemia dataset:

- Fig. 4.4 - Classification accuracy and chromosome length vs. number of generations for the leukaemia dataset using the length of the chromosome when calculating the values for the objective function (phase I). Using binary tournament selection.
- Fig. 4.5 - Classification accuracy and chromosome length vs. number of generations for the leukaemia dataset using the length of the chromosome when calculating the values for the objective function (phase I). Using rank based selection.
- Fig. 4.6 - Classification accuracy and chromosome length vs. number of generations for the leukaemia dataset without using the length of the chromosome when calculating the values for the objective function (phase I). Using binary tournament selection.
- Fig. 4.7 - Classification accuracy and chromosome length vs. number of generations for the leukaemia dataset without using the length of the chromosome when calculating the values for the objective function (phase I). Using rank based selection.

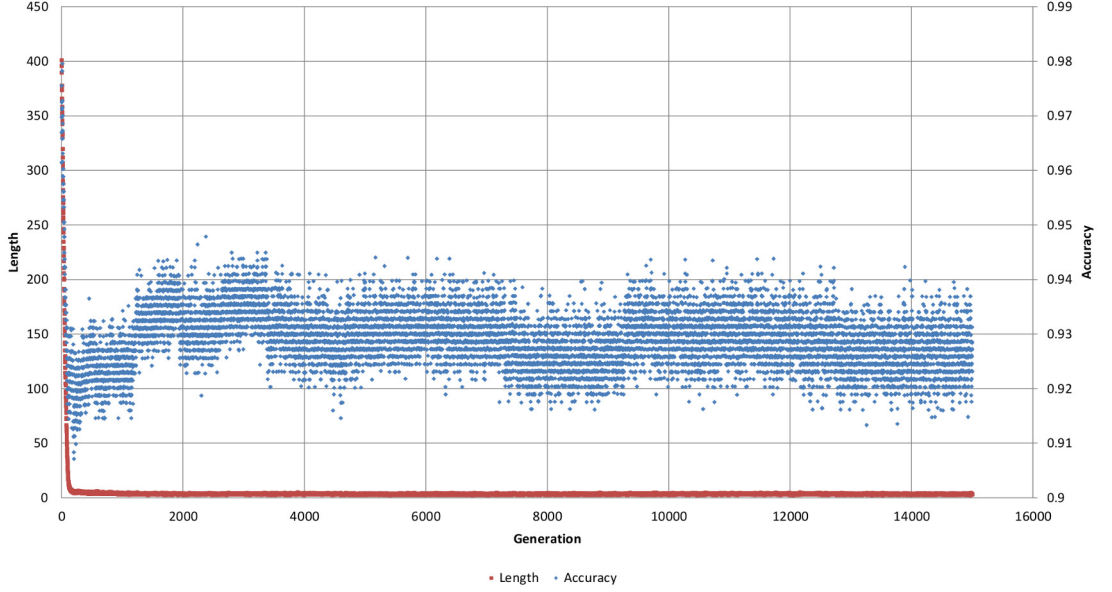


Figure 4.4: Leukaemia dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated taking the length of the chromosome into account and Binary Tournament Selection was used as the selection method.

As can be seen from these graphs for the leukaemia dataset, there is no clear end-point for phase one for the EA/ k -NN algorithm. An arbitrary end-point should not be used as this would hinder the qualitative comparisons that can be made with results from related literature.

Figures 4.8 & 4.9 illustrate this point further by looking at the same result for the ovarian cancer and prostate cancer datasets. The corresponding figure for the leukaemia dataset is Figure 4.4.

One other potential problem with the way that Juliusdottir et al. [38] set up the two-phase EA/ k -NN algorithm is model selection. They ran the algorithm in phase one mode for a pre-determined number of generations and then looked at the genes present in the final population. Unique genes present in the final populations from repeated runs of the algorithm were carried forward to the second phase of the algorithm. With this approach, it is possible that at least some of the informative genes may not be present in the final population. A better alternative would be to select the genes associated with the fittest chromosomes that appear across all the generations in phase one. These genes can then be taken over to phase two of the algorithm.

The new approach for selecting genes for phase two keeps track of all the chro-

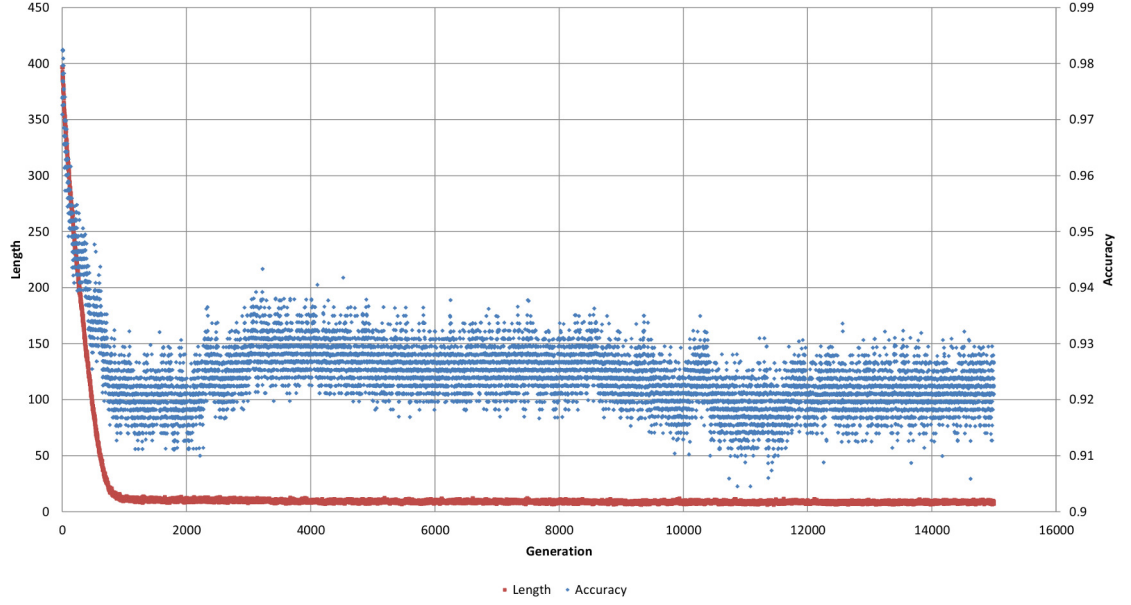


Figure 4.5: Leukaemia dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated without taking the length of the chromosome into account and Binary Tournament Selection was used as the selection method.

mosomes across all the generations in phase one. Then, for each chromosome in each generation, the following information is saved:

- The number of times a particular gene has appeared in a chromosome across all the generations (frequency of appearance)
- The sum of the values for the objective function evaluations for each chromosome that the gene appears in

Then, at the end, the algorithm calculates the average value for the objective function for each gene across all the generations by dividing the sum of the objective function values by the frequency of appearance. The top 10% of the genes are then carried over to the second phase of the algorithm.

This approach avoids both problems highlighted above. Furthermore, it has the advantage of dismissing certain genes regardless of their frequency of selection. This is because, despite the fact that some genes are selected frequently, they may act to lower the fitness of the chromosome that they are associated with. Such genes may be present in the final populations of phase one of the algorithm if it were set up in the way suggested by Juliusdottir et al. [38]. This is in part due to the fact that

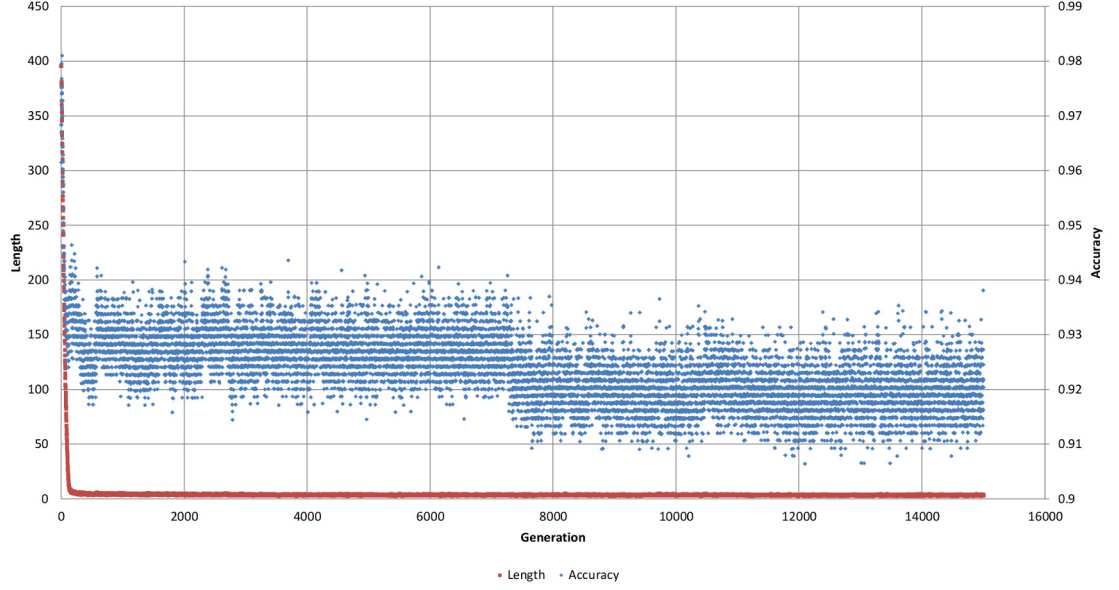


Figure 4.6: Leukaemia dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated taking the length of the chromosome into account and Rank Based Selection was used as the selection method.

phase one of the algorithm is not run for long enough for the algorithm to get to a point where the final population only contains informative genes.

As over-fitting is not taken into account during phase one, it may also lead to genes that have a negative effect on the classification performance of the chromosome being picked up for phase two. With the new approach, such genes will have a lower chance of getting through to the second phase of the algorithm.

The top 10% was selected as a reasonable figure by looking at the number of genes selected during phase one of the algorithm with the method suggested by Juliusdottir et al. [38].

4.7.4 The Set-Up of the Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm Using the Method Proposed in 4.7.3.1

As there was no clear way of determining the termination point for phase one of the two-phase EA/ k -NN algorithm, a novel method was proposed in section 4.7.3.1. This method was then used for testing the set-up of the two-phase EA/ k -NN algorithm the following way:

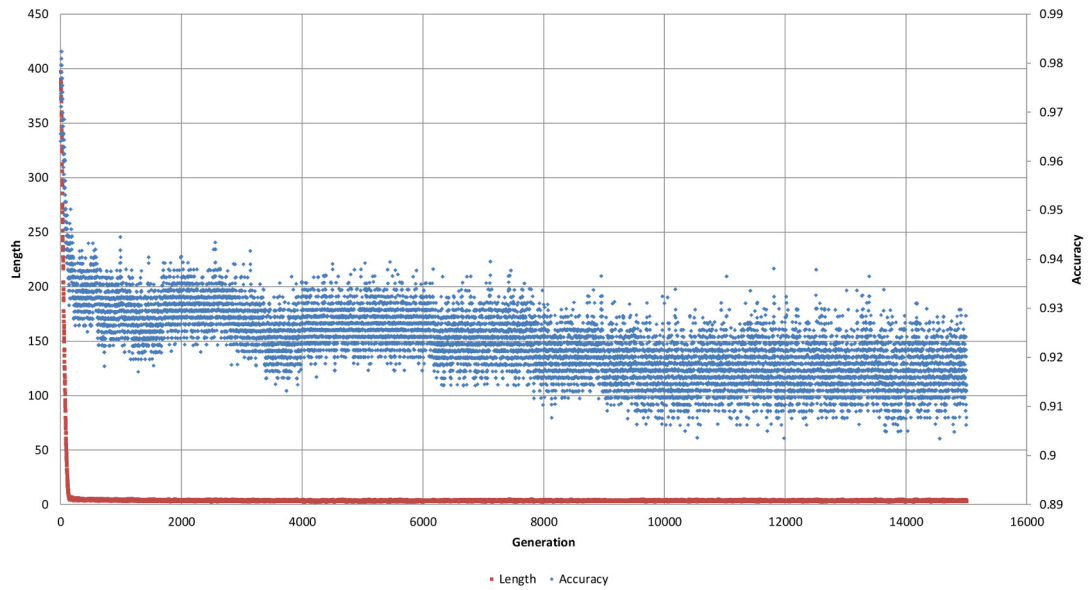


Figure 4.7: Leukaemia dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated without taking the length of the chromosome into account and Rank Based Selection was used as the selection method.

- BTS
 - P1W
 - P1WO
 - P1WO P2W
 - P1W P2W
- RBS
 - P1W
 - P1WO
 - P1WO P2W
 - P1W P2W

Where:

- P1W: Phase one run with the effect of the length of the chromosome taken into account
- P1WO: Phase one run without the effect of the length of the chromosome taken into account

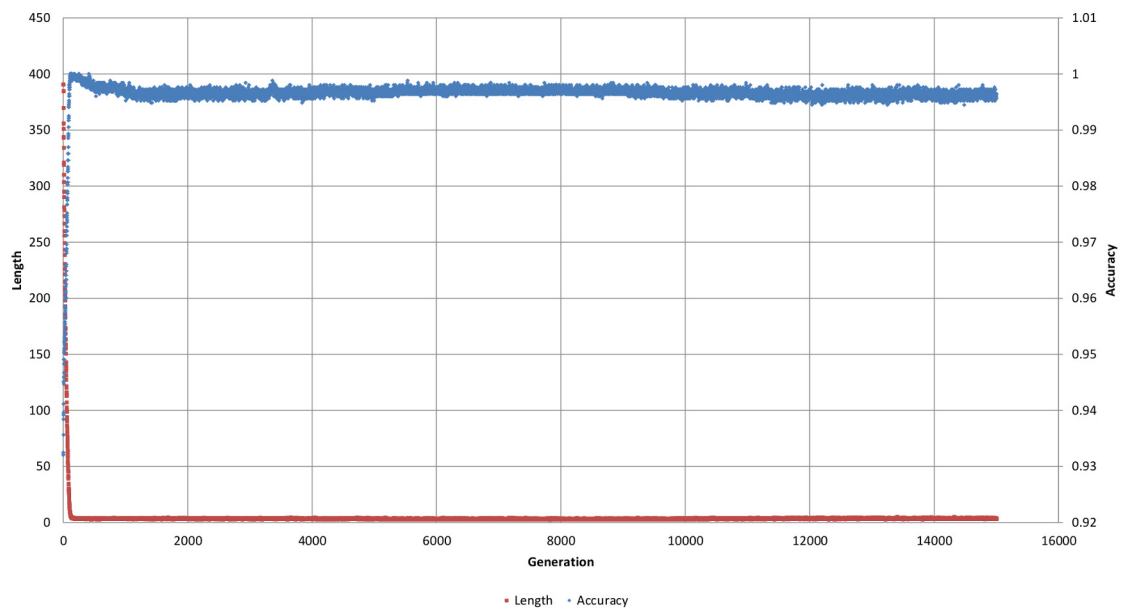


Figure 4.8: Ovarian cancer dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated taking the length of the chromosome into account and Binary Tournament Selection was used as the selection method.

- P2W: Phase two run with the effect of the chromosome taken into account

Each configuration of the algorithm was run for 10000 generations on the following datasets:

- Leukaemia
- Prostate cancer
- Colon cancer
- Breast cancer
- Ovarian cancer

Please refer to Chapter 1 for a brief overview of the datasets and Chapter 3 for a full explanation of the different ways to set-up the two-phase EA/ k -NN algorithm.

Table 4.8 shows the mean accuracy and Standard Deviation (SD) for the best chromosomes (selected models) over 8 repetitions for each method shown above. Table 4.9 shows the mean length and SD of the best validated chromosomes (selected models) for each method shown above.

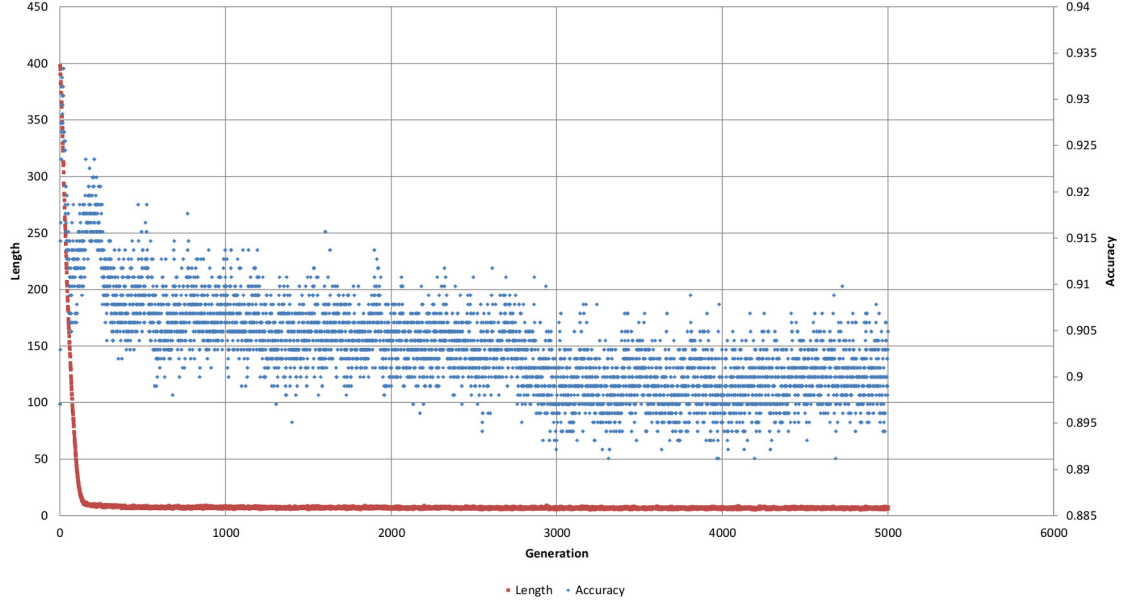


Figure 4.9: Prostate cancer dataset: classification accuracy and chromosome length vs. number of generations. The value of the objective function was calculated taking the length of the chromosome into account and Binary Tournament Selection was used as the selection method.

These methods were then analysed using ANOVA in order to determine the best method (or way of setting up the algorithm). The results from ANOVA are shown in Table 4.7. In this table, “rows” refer to datasets and “columns” refer to the different ways of setting up the two-phase EA/ k -NN algorithm.

ANOVA, developed by R. A. Fisher, is a process that aims to answer the question “Are there one or more significant differences *anywhere* among these samples?” [67]. It calculates an F -value using an F -test. The F -test compares the variability of values within a group to the variability of values between groups and produces an F -ratio (F -value). The F -value then needs to be checked against the F -distribution in order to reject the null hypothesis. The F -distribution is a continuous probability distribution whose shape depends on the number and the size of the samples [36, 67]. From the F -distribution applicable to the case under study, a critical F -value can be obtained at which the null hypothesis can be rejected.

The null hypothesis in this case is that all the means belong to the same population, therefore, there is no statistically significant difference between groups. The null hypothesis can be rejected once the probability of incorrectly rejecting it comes down to a certain significance level. This level is usually set to 5%. This means that there is a 5% chance of incorrectly rejecting the null hypothesis. In general, 5% is

considered to be an acceptable level for safely rejecting the null hypothesis.

As the F -value increases, the probability of incorrectly rejecting the null hypothesis decreases [67]. Common software packages that can carry out ANOVA (e.g. SPSS, Microsoft Excel etc.) calculate the F -value for the given data and output this F -value together with the critical F -value at the specified significance level (usually 5%). If the calculated F -value is higher than the critical F -value, then the null hypothesis can safely be rejected.

In Table 4.7, the F value for datasets is 268.61 and the critical F value is 2.71. Since the actual F value is higher than the critical F value, it can be concluded that there is a statistically significant difference between datasets. This is a logical conclusion as these datasets are completely different from each other, meaning that there should be marked differences in the way each machine learning method performs between these datasets.

The F value for each method is 1.02 and the critical F value is 2.36. This can be interpreted as there being no statistically significant difference between the various ways of setting up the algorithm across the datasets.

However, ANOVA, performed this way, is not a suitable statistical test for a few reasons. First of all, the usual assumption in ANOVA is that samples come from the same population. This clearly is not the case here. Samples (accuracies for each method on each dataset) come from different populations as the datasets are not related to each other. This fact has been verified by the first part of the statistical test itself.

Secondly, for results of ANOVA to hold, the underlying data must conform to some basic parameters. One of them is the “normality” of data. Normality of a distribution can be described by skewness and kurtosis [34]. Although skewness and kurtosis can describe a non-normal distribution, there is no general set of guidelines in this respect. This is due to the fact that skewness and kurtosis depend on sample size [53]. However, values for kurtosis that deviate significantly from zero can be taken as a good indicator of a non-normal distribution.

In general, a skewness value greater than zero indicates a right skewed distribution whereas a value less than zero indicates a left skewed distribution. A kurtosis value greater than zero indicates Leptokurtic distribution which is sharper than a normal distribution. In this type of distribution, values are concentrated around the mean and the tails are thicker. The implication is that there is a higher probability

for extreme values than in a normal distribution. A kurtosis value less than zero indicates a Platykurtic distribution which is flatter than a normal distribution with a wider peak. The implication in this case is that the probability for extreme values is less than for a normal distribution. The values are spread wider around the mean. Using these two metrics, it is possible to gain a good understanding about the underlying distribution. Therefore, descriptive statistics were carried out on the result set in order to ascertain the skewness and kurtosis values.

The result from this analysis is shown in Table 4.10. In this table, “columns” refer to the various ways in which the two-phase EA/ k -NN algorithm could be set-up. The skewness and kurtosis values indicate that the underlying distribution is a non-normal distribution.

In order to test this further, Shapiro-Wilk test [65] was carried out on the results set. The null hypothesis in this test is that the underlying distribution is a normal distribution. The P value returned from the test was 0.0130. Therefore, the null hypothesis can be rejected. This further confirms that the results set does not confirm to a normal distribution. Therefore, it can be concluded that ANOVA results are invalid. In general, if a few algorithms need to be compared across a few (non related) datasets, then it can be concluded that ANOVA cannot be used as the results returned by these datasets, as a whole, are unlikely to be normally distributed.

Therefore, in order to distinguish between the different ways of setting up the two-phase EA/ k -NN algorithm, a non-parametric, rank based method had to be used.

Mean ranks for each method were calculated as shown in Table 4.11. Then, a Java program was written that would randomize the ranking for each method for each dataset. A randomised mean rank for each method could then be calculated. The question to be answered was that if method 1 achieved an actual mean rank of 2.8, what would be the probability of this happening due to random chance? In order to answer this, the randomisation was repeated for 100000 times and the probability of a method achieving a mean ranking of 0.1 to 8.1 was plotted. This graph is shown in Figure 4.10.

Method 1 achieved an actual mean rank of 2.8. This corresponds to a probability of 0.13 of this result happening due to random chance. The next best method (method 2) achieved an actual mean rank of 3.2, which corresponds to a probability

of 0.23 of this result happening due to random chance. The worst method, method 5, achieved a mean rank of 6 which corresponds to a probability of 0.96 of it happening randomly. This clearly shows that some methods of setting up the algorithm are considerably better than others. In this case, method 1 (EA/ k -NN run in single phase mode with taking the length of the chromosome into account and using BTS as the selection method) seems to be the best way of running the EA/ k -NN algorithm. Therefore, this method will be used from here on.

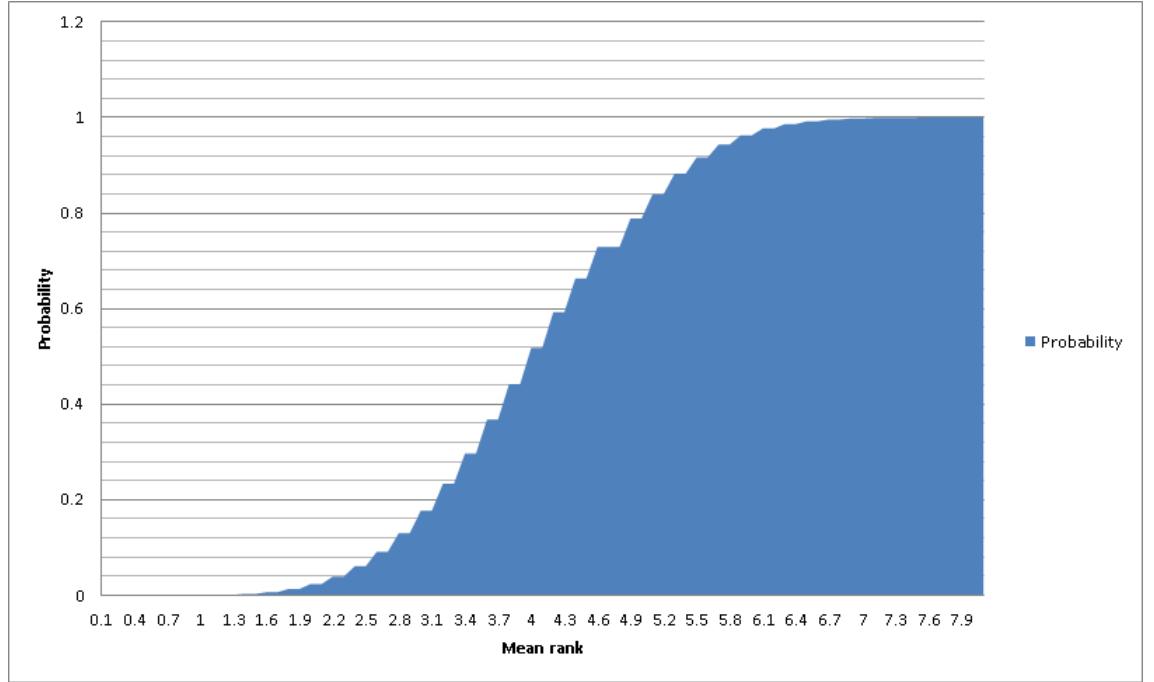


Figure 4.10: The probability of a mean rank happening by random chance.

4.7.5 Application of Feature Weights

As concluded in section 4.7.3.1, the first way (method 1) of setting up the two-phase EA/ k -NN algorithm is used here for the application of the weights. In this method, the algorithm is run in single phase mode (i.e. it is single phase rather than two-phase EA/ k -NN). The length of the chromosome is taken into account while evaluating the objective function and BTS is used as the selection method.

Table 4.4 summarises the cross-validated mean accuracy, mean length and associated SDs across eight repetitions for all the datasets for the EA/W- k -NN algorithm. In this algorithm, instead of classical k -NN, weighted k -NN is used as the classifier.

Table 4.5 summarises the cross-validated mean accuracy, mean length and associated SDs across eight repetitions for all the datasets for the version of the algorithm

| Dataset | Mean accuracy | SD | Mean length | SD |
|------------------------|---------------|------|-------------|------|
| Leukaemia | 0.79 | 0.10 | 1.25 | 0.46 |
| Prostate cancer | 0.85 | 0.06 | 3.63 | 1.30 |
| Breast cancer | 0.53 | 0.06 | 3.63 | 1.60 |
| Colon cancer | 0.72 | 0.06 | 2.75 | 1.75 |
| Ovarian cancer | 0.99 | 0.02 | 2.00 | 0.00 |

Table 4.4: Results from the application of feature weights to the Evolutionary Algorithm/ k -Nearest Neighbours algorithm.

| Dataset | Mean accuracy | SD | Mean length | SD |
|------------------------|---------------|------|-------------|------|
| Leukaemia | 0.90 | 0.09 | 1.88 | 0.99 |
| Prostate cancer | 0.81 | 0.06 | 6.13 | 8.54 |
| Breast cancer | 0.59 | 0.10 | 3.88 | 2.10 |
| Colon cancer | 0.81 | 0.06 | 2.75 | 1.98 |
| Ovarian cancer | 0.99 | 0.02 | 2.13 | 0.64 |

Table 4.5: Results from the Evolutionary Algorithm/Weighted Centroid Classification algorithm.

| Dataset | Mean accuracy | SD | Mean length | SD |
|------------------------|---------------|------|-------------|------|
| Leukaemia | 0.81 | 0.10 | 1.25 | 0.46 |
| Prostate cancer | 0.87 | 0.07 | 4.37 | 2.50 |
| Breast cancer | 0.58 | 0.11 | 9.87 | 9.70 |
| Colon cancer | 0.78 | 0.08 | 3.25 | 1.49 |
| Ovarian cancer | 0.98 | 0.02 | 2.00 | 0.53 |

Table 4.6: Baseline results to compare the two weighted algorithms against.

| SUMMARY | Count | Sum | Average | Variance |
|-----------------|-------|------|---------|----------|
| Leukaemia | 8 | 6.55 | 0.82 | 0.0010 |
| Prostate Cancer | 8 | 6.73 | 0.84 | 0.0005 |
| Breast Cancer | 8 | 4.50 | 0.56 | 0.0005 |
| Colon Cancer | 8 | 5.79 | 0.72 | 0.0014 |
| Ovarian Cancer | 8 | 7.79 | 0.97 | 0.0001 |
| Method 1 | 5 | 4.02 | 0.80 | 0.0218 |
| Method 2 | 5 | 3.94 | 0.79 | 0.0236 |
| Method 3 | 5 | 3.97 | 0.79 | 0.0198 |
| Method 4 | 5 | 3.93 | 0.79 | 0.0251 |
| Method 5 | 5 | 3.84 | 0.77 | 0.0197 |
| Method 6 | 5 | 3.88 | 0.78 | 0.0258 |
| Method 7 | 5 | 3.85 | 0.77 | 0.0269 |
| Method 8 | 5 | 3.91 | 0.78 | 0.0283 |

| ANOVA | | | | | |
|---------------------|-------|----|-------|--------|----------------|
| Source of Variation | SS | df | MS | F | P-value F crit |
| Datasets | 0.745 | 4 | 0.186 | 268.61 | 6.81E-022 2.71 |
| Methods | 0.005 | 7 | 0.001 | 1.02 | 0.44 2.36 |
| Error | 0.019 | 28 | 0.001 | | |
| Total | 0.770 | 39 | | | |

Table 4.7: Analysis of the various ways to set-up the two-phase Evolutionary Algorithm/ k -Nearest Neighbours algorithm using Analysis Of Variance. Please refer to 4.7.4 for an explanation of each method shown in this table.

| | Leukaemia | | Prostate Cancer | | Breast Cancer | | Colon Cancer | | Ovarian Cancer | |
|---------------|-----------|------|-----------------|------|---------------|------|--------------|------|----------------|------|
| | Mean | SD | Mean | SD | Mean | SD | Mean | SD | Mean | SD |
| BTS - P1W | 0.81 | 0.10 | 0.87 | 0.07 | 0.58 | 0.11 | 0.78 | 0.08 | 0.98 | 0.02 |
| BTS - P1WO | 0.84 | 0.08 | 0.83 | 0.08 | 0.58 | 0.10 | 0.71 | 0.12 | 0.99 | 0.02 |
| BTS - P1WOP2W | 0.76 | 0.10 | 0.88 | 0.07 | 0.59 | 0.05 | 0.77 | 0.09 | 0.97 | 0.02 |
| BTS - P1WP2W | 0.85 | 0.08 | 0.83 | 0.12 | 0.56 | 0.10 | 0.71 | 0.09 | 0.98 | 0.02 |
| RBS - P1W | 0.81 | 0.10 | 0.82 | 0.06 | 0.58 | 0.08 | 0.69 | 0.08 | 0.95 | 0.05 |
| RBS - P1WO | 0.82 | 0.10 | 0.85 | 0.05 | 0.56 | 0.06 | 0.68 | 0.09 | 0.98 | 0.02 |
| RBS - P1WOP2W | 0.80 | 0.10 | 0.83 | 0.12 | 0.53 | 0.11 | 0.73 | 0.12 | 0.98 | 0.02 |
| RBS - P1WP2W | 0.86 | 0.09 | 0.82 | 0.08 | 0.53 | 0.08 | 0.72 | 0.09 | 0.98 | 0.02 |

Table 4.8: Mean classification accuracy and Standard Deviation for different ways of setting up the two-phase Evolutionary Algorithm/ k -Nearest Neighbours algorithm across five datasets.

| | Leukaemia | | Prostate Cancer | | Breast Cancer | | Colon Cancer | | Ovarian Cancer | |
|---------------|-----------|------|-----------------|------|---------------|------|--------------|------|----------------|------|
| | Mean | SD | Mean | SD | Mean | SD | Mean | SD | Mean | SD |
| BTS - P1W | 1.25 | 0.46 | 4.38 | 2.50 | 9.88 | 9.70 | 3.25 | 1.49 | 2.00 | 0.53 |
| BTS - P1WO | 1.88 | 0.83 | 6.38 | 2.45 | 5.25 | 1.16 | 5.75 | 8.73 | 2.00 | 0.00 |
| BTS - P1WOP2W | 1.50 | 0.53 | 4.13 | 2.23 | 5.25 | 2.25 | 2.63 | 1.19 | 1.88 | 0.35 |
| BTS - P1WOP2W | 1.38 | 0.52 | 2.88 | 0.83 | 3.75 | 1.58 | 2.50 | 1.20 | 1.88 | 0.35 |
| RBS - P1W | 1.25 | 0.46 | 3.38 | 1.06 | 4.50 | 2.07 | 2.75 | 1.49 | 1.88 | 0.35 |
| RBS - P1WO | 1.38 | 0.74 | 3.75 | 1.91 | 4.63 | 2.50 | 2.38 | 1.19 | 2.00 | 0.00 |
| RBS - P1WOP2W | 1.25 | 0.46 | 4.13 | 2.30 | 4.88 | 1.73 | 3.75 | 2.71 | 1.88 | 0.35 |
| RBS - P1WOP2W | 1.38 | 0.52 | 3.38 | 1.41 | 3.38 | 1.60 | 2.50 | 0.93 | 1.88 | 0.35 |

Table 4.9: Mean chromosome length and Standard Deviation for different ways of setting up the two-phase Evolutionary Algorithm/ k -Nearest Neighbours algorithm across five datasets.

| | BTS | | BTS | | BTS | | RBS | | RBS | | RBS | |
|--------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | P1W | P1WO | P1WOP2W | P1WOP2W | P1W | P1WO | P1W | P1WO | P1WOP2W | P1WOP2W | P1W | P1WO |
| Mean | 0.80 | 0.79 | 0.79 | 0.79 | 0.77 | 0.78 | 0.77 | 0.78 | 0.77 | 0.77 | 0.77 | 0.78 |
| Standard Error | 0.07 | 0.07 | 0.06 | 0.06 | 0.06 | 0.07 | 0.06 | 0.07 | 0.07 | 0.07 | 0.07 | 0.08 |
| Median | 0.81 | 0.83 | 0.77 | 0.77 | 0.81 | 0.82 | 0.81 | 0.82 | 0.80 | 0.80 | 0.80 | 0.82 |
| Standard Deviation | 0.15 | 0.15 | 0.14 | 0.14 | 0.14 | 0.16 | 0.14 | 0.16 | 0.16 | 0.16 | 0.16 | 0.17 |
| Sample Variance | 0.02 | 0.02 | 0.02 | 0.02 | 0.02 | 0.03 | 0.02 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 |
| Kurtosis | 1.54 | 0.03 | 0.45 | 0.45 | -0.15 | -0.71 | -0.15 | -0.71 | 1.25 | 1.25 | 0.47 | 0.47 |
| Skewness | -0.80 | -0.26 | -0.40 | -0.40 | -0.27 | -0.24 | -0.27 | -0.24 | -0.55 | -0.55 | -0.65 | -0.65 |
| Range | 0.40 | 0.41 | 0.38 | 0.38 | 0.37 | 0.42 | 0.37 | 0.42 | 0.45 | 0.45 | 0.45 | 0.45 |
| Minimum | 0.58 | 0.58 | 0.59 | 0.59 | 0.58 | 0.56 | 0.58 | 0.56 | 0.53 | 0.53 | 0.53 | 0.53 |
| Maximum | 0.98 | 0.99 | 0.97 | 0.97 | 0.95 | 0.98 | 0.95 | 0.98 | 0.98 | 0.98 | 0.98 | 0.98 |
| Sum | 4.02 | 3.94 | 3.97 | 3.97 | 3.84 | 3.88 | 3.84 | 3.88 | 3.85 | 3.85 | 3.91 | 3.91 |
| Count | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |

Table 4.10: Descriptives statistics on the results of various ways of setting up the two-phase Evolutionary Algorithm/ k -Nearest Neighbours algorithm across five datasets.

that uses the weighted centroid classification approach.

The baseline against which the performance of both the algorithms is measured is the EA/ k -NN algorithm with classical k -NN running in single phase mode with BTS as the selection method. Results for the baseline method are shown in Table 4.6.

Figure 4.11 shows this information in graphical form. The EA/W- k -NN algorithm performs worse than the baseline on all the datasets apart from the ovarian cancer dataset, whereas the weighted centroid classifier algorithm performs better than the baseline on all the datasets apart from the prostate cancer dataset. From these results, it can be concluded that the weighted centroid classification algorithm proposed here looks promising for feature selection and classification in predictive data mining.

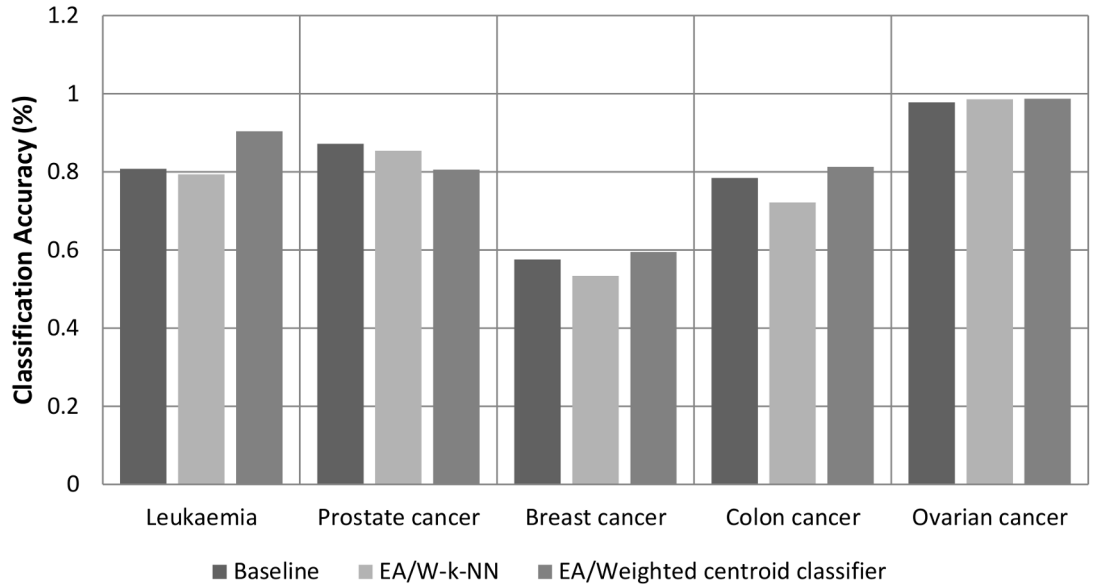


Figure 4.11: Comparison of the baseline method vs. weighted methods.

4.7.6 Conclusion

The first part of this chapter focused on setting up the two-phase EA/ k -NN algorithm in the best possible way across a number of datasets. This was done so that a baseline performance measure could be calculated for the best method for each dataset used here.

An attempt was made to use classical statistics (e.g. ANOVA) to analyse the results for the different ways of setting up the algorithm across a few datasets. However, due to the nature of the problem at hand, it turned out that classical

| | Leukaemia | | Prostate Cancer | | Breast Cancer | | Colon Cancer | | Ovarian Cancer | | Mean Rank |
|---------------|-----------|---|-----------------|---|---------------|---|--------------|---|----------------|---|-------------|
| | Rank | | Rank | | Rank | | Rank | | Rank | | Rank |
| BTS - P1W | 0.81 | 5 | 0.87 | 2 | 0.58 | 2 | 0.78 | 1 | 0.98 | 4 | 2.80 |
| BTS - P1WO | 0.84 | 3 | 0.83 | 5 | 0.58 | 2 | 0.71 | 5 | 0.99 | 1 | 3.20 |
| BTS - P1WOP2W | 0.76 | 8 | 0.87 | 1 | 0.59 | 1 | 0.77 | 2 | 0.97 | 7 | 3.80 |
| BTS - P1WP2W | 0.85 | 2 | 0.83 | 4 | 0.56 | 5 | 0.71 | 5 | 0.98 | 3 | 3.80 |
| RBS - P1W | 0.81 | 5 | 0.82 | 8 | 0.58 | 2 | 0.69 | 7 | 0.95 | 8 | 6.00 |
| RBS - P1WO | 0.82 | 4 | 0.85 | 3 | 0.56 | 5 | 0.68 | 8 | 0.98 | 5 | 5.00 |
| RBS - P1WOP2W | 0.80 | 7 | 0.83 | 6 | 0.53 | 8 | 0.73 | 3 | 0.98 | 5 | 5.80 |
| RBS - P1WP2W | 0.86 | 1 | 0.82 | 7 | 0.53 | 7 | 0.72 | 4 | 0.98 | 2 | 4.20 |

Table 4.11: Calculation of mean ranks for each method of setting up two-phase Evolutionary Algorithm/ k -Nearest Neighbours algorithm.

statistics may not be able to provide a conclusion.

Therefore, a non-parametric, randomised ranking based method was used for determining the best way of setting up the two-phase EA/ k -NN algorithm. In contrast to Chapter 3, results from this study indicated that the single phase EA/ k -NN outperformed the two-phase EA/ k -NN algorithm.

Single phase EA/ k -NN was then tested with adaptive weights (W- k -NN) in order to determine if weights could improve the performance of the algorithm. Initial testing strongly indicated that with the correct value of T for the adaptive weights, W- k -NN is able to outperform the classical k -NN algorithm.

However, further testing in combination with an EA showed the adaptive weights scheme to be counterproductive to the k -NN algorithm.

After that, a novel way of classifying samples was tested across the five datasets. In this method, centroid classification was used instead of classical k -NN. However, instead of normal Euclidean distances, weighted (adaptive weights) Euclidean distances were used in centroid classification. This method outperformed classical k -NN in all the datasets apart from one.

Therefore, the weighted centroid classifier looks promising for feature selection and classification in predictive data mining.

Chapter 5

Correlation Guided Evolutionary Algorithm/ k -Nearest Neighbours for Feature Selection

5.1 Introduction

The two-phase EA/ k -NN algorithm introduced by Juliusdottir et al. [38] was extensively studied for FS in five large biomedical datasets in Chapter 4. It was concluded in Chapter 4 that single phase EA/ k -NN was better at FS and classification across the datasets tested compared to the two-phase EA/ k -NN algorithm.

From the experiments conducted in Chapter 4, it was clear that the selection and retention of information rich features is of paramount importance to FS and classification in noisy, high dimensional datasets such as DNA microarray data.

Statistical correlation is used widely in FS. Some techniques rank genes using correlation as a way of prior-selection before applying machine learning algorithms to the dataset [68, 66, 13, 83]. It is claimed that prior-selection considerably improves the performance of machine learning algorithms [38]. Hall and Smith [27] demonstrated that prior feature selection using a correlation based filter approach significantly increased the performance of three machine learning algorithms.

This indicates that feature correlation carries knowledge and this approach should improve the performance of an EA as well. Therefore, it is logical to assume that some degree of improvement could be made to the EA/ k -NN algorithm if the search could be guided with information gained by correlation coefficients calculated for

features in the dataset.

Correlation between features can be quantified using different coefficients. Pearson's coefficient (r), Spearman's rho coefficient (r_s), and Kendall's tau (τ) coefficient are three of the most popular correlation coefficients [28].

Pearson's correlation coefficient is a parametric statistical technique that can capture linear relationships between two features. It assumes that the underlying data belongs to a normal distribution and it is applied to raw data [67]. Pearson's correlation coefficient can be calculated using Equation 5.2.

The Spearman's r_s is a rank based correlation coefficient introduced by Spearman [74]. It is generally expressed as [84]:

$$r_s = 1 - 6 \sum d^2 / (n^3 - n) \quad (5.1)$$

Where:

- n is the number of measurements
- $\sum d^2 = \sum_{i=1}^n d_i^2$
- d_i is the ranked difference between i th measurements

Spearman's rank correlation coefficient is a nonparametric technique that is calculated on ranked data. As this is a nonparametric technique, it does not depend on the underlying population conforming to a normal distribution. As it operates on ranked data, it is also more tolerant to outliers. However, there is a loss of information when raw data is converted to ranks. Therefore, for a normally distributed dataset, Pearson's correlation coefficient is more powerful than Spearman's [20].

Kendall's τ , introduced by Kendall in 1938 [39] is a correlation coefficient that can be used as an alternative to Spearman's r_s . Kendall's τ is a rank based coefficient. It is described as a simple function of the minimum number of neighbour swaps needed to produce one ordering from another by Hauke and Kossowski [28]. Kendall argued that although Spearman's r_s is easier to calculate than τ , from a theoretical point of view, τ is preferable to r_s [39].

After carrying out statistical significance testing on both Pearson's and Spearman's correlation coefficients, Hauke and Kossowski concluded that it is possible to find situations where Pearson's coefficient is negative while Spearman's is positive.

However, they argue that results from Spearman's rank correlation coefficient should not be over interpreted as a measure of significance between two variables [28].

Although, in theory, Kendall's τ is preferable to Spearman's rank correlation coefficient, as τ is harder to calculate and not commonly used, it was decided that τ will not be used in this thesis. Therefore, the choice of correlation coefficient was between Pearson's and Spearman's. It was decided that this thesis would investigate Pearson's correlation coefficient as it is potentially more powerful compared to Spearman's.

This decision was made due to the fact that the point of introducing correlation coefficient into the EA/ k -NN is to help guide the search process towards promising feature subsets. The EA/ k -NN algorithm has been proven to be a powerful feature selection and classification tool in Chapter 3 and Chapter 4 of this thesis. Therefore, it can be argued that the correlation coefficient that is most likely to extract useful information from the dataset is more likely to fit the requirements. As Spearman's rank correlation coefficient considers ranks instead of raw data and therefore loses some information in the process, it was decided that Pearson's correlation coefficient would be a better fit here.

In the context of feature selection, statistical correlation could be used in two different ways:

- Include the most strongly correlated features in the selected feature subset.
- Include a diverse set of loosely correlated features in the selected feature subset.

Common techniques that use correlation coefficient for feature selection generally prefer strongly correlated feature subsets [68, 66, 13, 83]. However, there is evidence in literature [43, 42, 16] that supports preferring loosely correlated features in the selected feature subset. The rationale for this argument is that for efficient feature selection, irrelevant features should be eliminated (in addition to redundant features). A feature is regarded as redundant if it is strongly correlated to an already selected feature with a high classification performance. The redundancy arises from the fact that each feature on its own is capable of high classification performance and therefore, combining these features does not increase the knowledge contained within the model. This theory could be applied in this case by changing the selection bias towards loosely correlated features. In theory, this would give the EA a

wider search area and it may force the EA to select a feature subset that is highly correlated to the case under study.

This chapter investigates both correlation based approaches to feature selection using Pearson's product-moment correlation coefficient [10].

5.2 Overview of the Algorithm

The same algorithm that was used in Chapter 4 was used as the basis here (please see 4.4.2 for more details). This algorithm is a single objective, generational, elitist EA that uses k -NN as the classifier/objective function. Chromosomes in the EA are integer encoded and variable in length. Genes in the chromosomes range from 1 to n where n is the number of features in the dataset.

The algorithm performs the following steps during a typical run in single phase mode:

- Read the training fold of the dataset from disk
- Create a population store (the population store tracks chromosomes across generations)
- Create the initial random population and add it to the population store
- Calculate the values for the objective function for all the chromosomes in the initial population
- Add the initial population to the population store
- Repeat for *NUM_GENERATIONS*:
 - Select (*POP_SIZE* - *ELITIST_COUNT*) pairs of parents
 - Apply crossover operator to produce a child from each pair of parents
 - Apply mutation operator to each child (33% of the time add a random gene, 33% of the time remove a random gene, 33% of the time change a random gene to another random gene)
 - Calculate the values for the objective function for each child
 - Replace parents (except the elitist parents) with children

- Read the test fold of the dataset
- Calculate the test objective function values for all the chromosomes across all the generations in the population store
- Mark the top chromosome/s for each generation depending on the test accuracy and the length of the feature subset encoded by the chromosome
- Read the validation fold of the dataset
- Validate only the chromosomes marked as top using validation data

5.3 Correlation Guided Mutation

Statistical correlation is used widely in FS. However, it is not used frequently together with EAs for FS and classification. It was hypothesised that correlation coefficient could provide additional information to the EA and therefore make the search for the optimal feature subset more effective.

The most logical place in the EA where correlation coefficient could be introduced is the mutation operator. This is due to the fact that the mutation operator directly adds, removes or modifies genes in the chromosome. Therefore, the mutation operator could be influenced by the correlation coefficient of the features in the dataset.

In order to achieve this, a method that calculated the correlation coefficient matrix for the training fold of the dataset was incorporated to the algorithm immediately after reading the training fold of the dataset from disk. This method calculates the correlation coefficient matrix on the entire training fold of the dataset. It then calculates the mean correlation coefficient for each feature in the training fold as shown in Tables 5.1 and 5.2. Please see section 5.4 for an explanation of the method used for calculating the correlation coefficient. The mean is calculated using the absolute value of the correlation coefficient as both positively and negatively correlated features are of interest to the case under study.

| | F1 | F2 | F3 | F4 | F5 | F6 | Classification |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------------------|
| S1 | 1 | 8 | 3 | 6 | 10 | 2 | Normal |
| S2 | 5 | 1 | 8 | 5 | 5 | 5 | Normal |
| S3 | 4 | 8 | 5 | 8 | 1 | 5 | Normal |
| S4 | 6 | 1 | 6 | 5 | 8 | 3 | Normal |
| S5 | 3 | 2 | 9 | 4 | 4 | 2 | Cancer |
| S6 | 4 | 9 | 6 | 5 | 10 | 6 | Cancer |
| S7 | 9 | 4 | 10 | 10 | 9 | 9 | Cancer |

Table 5.1: An artificial dataset for illustrating the calculation of mean correlation coefficient for each feature. This dataset contains seven samples (S1 - S7). Each sample has six features F1 - F6).

| | F1 | F2 | F3 | F4 | F5 | F6 |
|------------------------|-----------|-----------|-----------|-----------|-----------|-----------|
| F1 | 1.00 | 0.41 | 0.69 | 0.58 | 0.10 | 0.79 |
| F2 | 0.41 | 1.00 | 0.61 | 0.27 | 0.18 | 0.13 |
| F3 | 0.69 | 0.61 | 1.00 | 0.17 | 0.13 | 0.50 |
| F4 | 0.58 | 0.27 | 0.17 | 1.00 | 0.01 | 0.74 |
| F5 | 0.10 | 0.18 | 0.13 | 0.01 | 1.00 | 0.16 |
| F6 | 0.79 | 0.13 | 0.50 | 0.74 | 0.16 | 1.00 |
| Mean (absolute) | 0.59 | 0.44 | 0.52 | 0.46 | 0.26 | 0.55 |

Table 5.2: Correlation coefficient matrix and the mean correlation coefficients for each feature for the dataset shown in 5.1.

5.3.1 Correlation Guided Mutation - Remove a Gene

5.3.1.1 Preference Towards Selecting Strongly Correlated Features

In order to remove a gene from a chromosome using correlation information, the genes encoded in the chromosome are sorted into descending order by their mean correlation coefficient. After sorting, the genes that have the highest mean correlation coefficients are at the beginning of the chromosome and the genes with the lowest correlation coefficients are at the end of the chromosome. A biased random number (please see section 5.5) is then generated between 0 and $CHROMOSOME_LENGTH - 1$. The gene pointed to by this random number is then removed from the chromosome. As this number is biased towards $CHROMOSOME_LENGTH - 1$, this makes sure that genes with lower mean correlation coefficients have a higher chance of being removed.

5.3.1.2 Preference Towards Selecting Loosely Correlated Features

This approach for removing a gene operates in an identical fashion to the approach described in section 5.3.1.1, apart from one difference: instead of using a descending order, the genes encoded in the chromosome are sorted into ascending order by their mean correlation coefficient. As the random number that is generated is biased towards $CHROMOSOME_LENGTH - 1$, this ensures that most correlated genes in the chromosome have a higher chance of being removed.

5.3.2 Correlation Guided Mutation - Add a Gene

5.3.2.1 Preference Towards Selecting Strongly Correlated Features

Adding a correlation guided gene works in a similar manner to removing a correlation guided gene. However, in this case, rather than looking at the mean correlation coefficients of the genes in the chromosome, the correlation coefficients of the genes present in the entire training set are looked at. The features of the dataset are sorted into ascending order by their mean correlation coefficients. A biased random number is used for selecting a gene to be added to the chromosome. As the biased random number is biased towards larger numbers and strongly correlated features of the dataset are towards the end (after sorting into ascending order), this ensures that highly correlated genes have a better chance of being included in the chromosome.

A constraint is applied to this process to make sure that the gene that is being added is not already present in the chromosome.

5.3.2.2 Preference Towards Selecting Loosely Correlated Features

This approach operates in an identical fashion to the approach described in section 5.3.2.1, apart from one difference: the features of the dataset are sorted into descending order by their correlation coefficients. After sorting, loosely correlated features are placed towards the end of the dataset. As the biased random number is biased towards larger numbers, this, together with the sorting of the dataset, ensures that loosely correlated genes have a higher chance of being added to the chromosome.

5.3.3 Correlation Guided Mutation - Change a Gene

5.3.3.1 Preference Towards Selecting Strongly Correlated Features

This again is similar to correlation guided removal and addition of a gene. In this case, the correlation coefficients of the genes in the chromosome and also the entire training fold are looked at. First of all, a gene from the chromosome is selected for changing. This is achieved by sorting the genes in the chromosome into descending order by correlation coefficient and selecting a gene using a biased random number. This ensures that genes with lower correlation coefficients have a better chance of being changed.

Then, the correlation coefficients for the dataset are sorted into ascending order and a gene is picked using a biased random number so that genes that are highly correlated have a better chance of being selected.

A constraint is applied to this process to make sure that the gene that is being changed does not change to a gene that is already present in the chromosome.

5.3.3.2 Preference Towards Selecting Loosely Correlated Features

This approach works in a similar manner to the approach described in section 5.3.3.1, apart from one difference. In the approach outlined in section 5.3.3.1, genes encoded in the chromosome and the features of the dataset are sorted by their correlation coefficients so that the least correlated genes in the chromosome have a higher chance of being changed into the most correlated features in the dataset. In this approach, the sort orders are changed so that the most correlated genes in the chromosome have a higher chance of being replaced with the least correlated features in the dataset.

5.4 Calculation of the Correlation Coefficient Matrix

Equation 5.2 [45] was used for calculating the correlation coefficient between two features.

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2}} \quad (5.2)$$

Where:

- x_i = feature x from i th sample
- y_i = feature y from i th sample
- \bar{x} = mean of feature x
- \bar{y} = mean of feature y

Equation 5.2 was interpreted in the form of the following algorithm in order to calculate the correlation coefficient between two features (x and y) of the training fold:

- For each sample in the training fold:
 - Sum feature x
 - Sum feature y
- Calculate the means for features \bar{x} and \bar{y}
- For each sample in the training fold:
 - Calculate $(x - \bar{x}) * (y - \bar{y})$ and add this to a running total named $sumXY$
 - Calculate $(x - \bar{x})^2$ and add this to a running total named $sumX^2$
 - Calculate $(y - \bar{y})^2$ and add this to a running total named $sumY^2$
- Calculate the coefficient by using $sumXY / (\sqrt{sumX^2} * \sqrt{sumY^2})$

5.5 Biased Random Number Generation

Java provides built-in methods for generating uniformly distributed random numbers between any range. However, there is no built-in method for generating random numbers within a specified range with a bias towards one end of the range. The availability of such a random number generator is a requirement for the correlation guided mutation operator described in section 5.3.

For example, if a chromosome encodes 10 features, then it is necessary to generate a random number between 0 and 9 that is biased towards 9. Then, by sorting the chromosome into descending order by the magnitude of the correlation coefficient of each gene, this random number is more likely to point to a gene that has a relatively low correlation coefficient.

Such a random number can be generated by using the following procedure:

- Generate a random number between 0 and 1, let this be R
- Transform the random number using $R = R^{bias}$ where *bias* is an adjustable parameter
- Scale the random number to a custom range using $R = R * (1 + (TOP - 1))$ where *TOP* is the number of features in a chromosome
- Return the rounded R as an integer

Figure 5.1 shows the frequency of random numbers generated using the above procedure with the following parameters:

- Total number of random numbers generated = 10000
- $bias = 0.5$
- $TOP = 10$

Figure 5.2 shows another set of 10000 biased random numbers with a *bias* of 0.25. From these two graphs, it is clear that a *bias* of 0.5 is more appropriate compared to a *bias* of 0.25. With a *bias* of 0.25, genes with a very low correlation coefficient have no chance of being selected. This may be counterproductive and goes against one of the core principles of EAs. In an EA, the fittest individuals have a better chance of being selected as parents and therefore producing children. However, even

individuals with lower fitness values do have a chance of being selected although the chance is very low compared to the fittest individuals. Therefore, in this instance, it is logical to use a value for *bias* that gives a chance to even the least correlated genes.

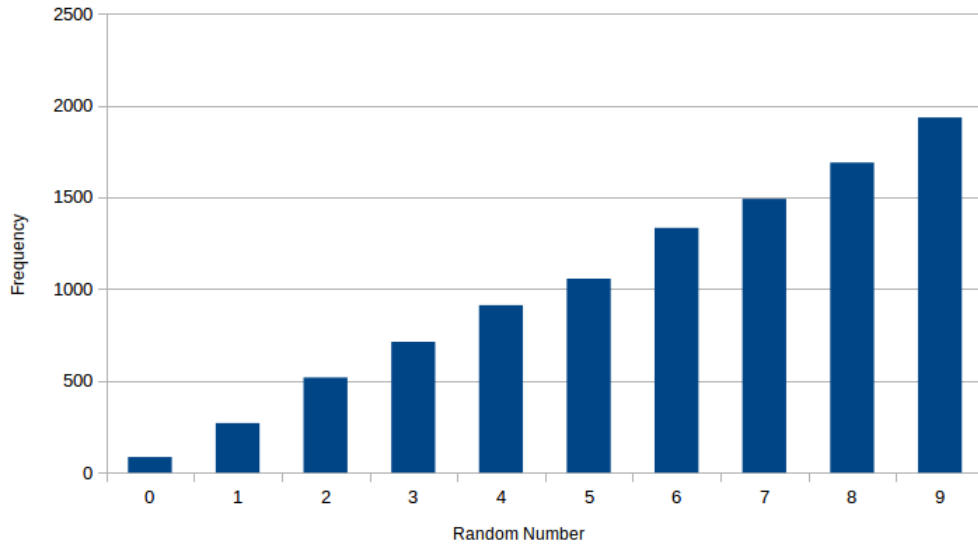


Figure 5.1: Frequency of 10000 biased random numbers with a *bias* of 0.5.

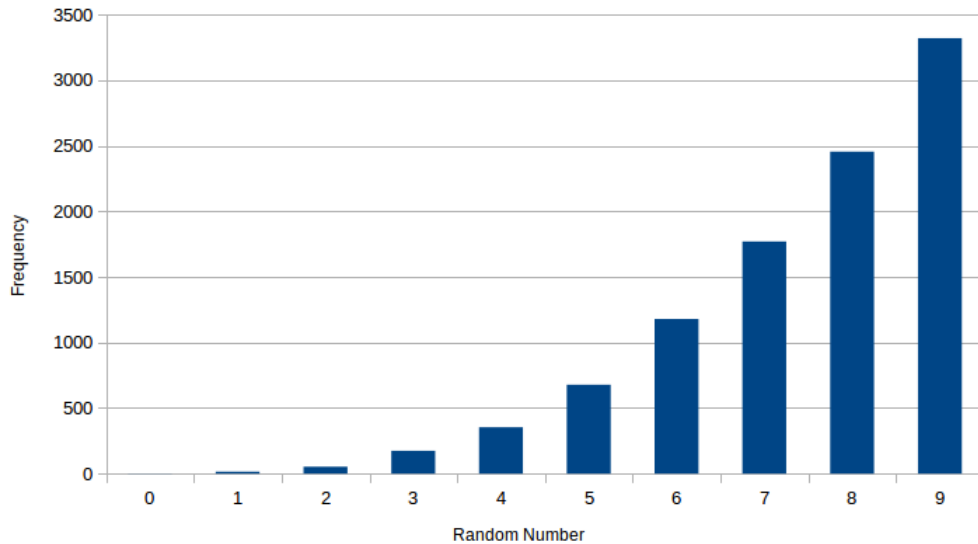


Figure 5.2: Frequency of 10000 biased random numbers with a *bias* of 0.25.

5.6 Experiment Set-up

All 5 datasets (please refer to section 1.3 for details on the datasets) were tested using EA/ k -NN algorithm with correlation coefficient guided mutation. Experiments were set up with the following parameters:

- Number of generations = 10000
- Population size = 50
- Initial chromosome size = 30
- Selection method = BTS
- Use chromosome length in the objective function = true
- $\alpha = 1000$
- Algorithm mode = single phase

Eight separate runs were carried out on each of the datasets. For each run, the dataset was randomly split into three stratified folds. The algorithm was trained on the training fold. Then, all the chromosomes across all the generations were tested on the testing fold (unseen data). For each generation, the top chromosome was marked as the selected model. The selected models were then validated on the validation fold (unseen data).

Then, as a post processing step, the best testing chromosome across all the generations (the best chromosome is defined as the chromosome with the highest testing accuracy and the lowest feature subset at the earliest generation) was selected and the validation accuracy of this chromosome was noted. The reported validation accuracy then is the average validation accuracy across the eight separate runs.

Initially, two batches of experiments were carried out using EA/ k -NN algorithm (without weights):

- Most correlated approach: In this set of experiments, the correlation guided mutation added, removed or changed genes in the chromosome (selected feature subset) with a preference towards strongly correlated genes being included in the chromosome.

- Least correlated approach: In this set of experiments, the mutation operator added, removed or changed genes in the chromosome with a preference towards having loosely correlated set of genes.

Results were then analysed and the most promising approach (the most correlated approach) was then applied to the best approach from Chapter 4 (EA/weighted centroid classifier).

5.7 Results and Discussion

Tables 5.7 and 5.7 show overall results across the five datasets for EA/ k -NN algorithm with correlation guided mutation for the most correlated approach and least correlated approach respectively. Table 5.7 shows results across all five datasets for the correlation guided mutation approach with a preference for strongly correlated features applied to the EA/weighted centroid classifier algorithm. Then, Table 5.6 shows the results from the three sets of correlation guided experiments in comparison to the baseline algorithm, W- k -NN algorithm and EA/weighted centroid classification algorithm.

As shown in Table 5.6, the correlation guided mutation approach with a preference for strongly correlated feature subsets achieves slightly better results on two out of the five datasets (prostate & ovarian cancer datasets). The correlation guided mutation approach with a preference for loosely correlated feature subsets only achieved better results on one out of the five datasets (colon cancer dataset). When combined, correlation guided approaches achieved better results than the other approaches shown in Table 5.6 in three out of the five datasets.

When compared with each other, from the two correlation guided mutation approaches, the approach with preference for strongly correlated feature subsets achieves better results on two out of the five datasets (prostate and ovarian cancer datasets) while the approach with a preference for loosely correlated feature subsets achieves better results on one out of the five datasets (colon cancer dataset). On two out of the five datasets (leukaemia and breast cancer datasets), both approaches achieved the same result. As the mutation guided approach with a preference for strongly correlated feature subsets achieved better results than the approach with a preference for loosely correlated feature subsets, this approach was then tested with

the EA/Weighted Centroid algorithm from Chapter 4.

The EA/Weighted Centroid algorithm, together with the correlation guided mutation that had a preference for strongly correlated subset of features, achieved better (or joint best) results for four out of the five datasets. The only dataset that this approach failed to achieve the best results for was the prostate cancer dataset.

This result further demonstrates that the EA/Weighted Centroid algorithm introduced in Chapter 4 is capable of outperforming the algorithms used in this thesis across the datasets tested in this thesis.

If only the EA/Weighted Centroid is analysed before and after applying the correlation guided mutation approach, then EA/Weighted Centroid shows improved results on three out of the five datasets (leukaemia, prostate cancer and colon cancer datasets) after applying the correlation guided mutation approach. On the other two datasets, the correlation guided mutation approach did not make any difference to the end result. This clearly indicates that the correlation guided mutation approach with a preference towards strongly correlated features in the selected feature subset is capable of improving the performance of the EA/Weighted Centroid algorithm across the five datasets that were tested.

When the correlation guided mutation approach with a preference towards selected loosely correlated features is compared to the baseline, it manages to achieve better results on one out of the five datasets while achieving slightly worse results on two out of the five datasets. This indicates that even when the mutation operator is forcing the EA to search for loosely correlated features, which may point the EA towards noisy features, the EA is still capable of selecting feature subsets with good classification performance. This is an indication of the robustness of the EA in feature selection.

However, there may be room for improvement. First of all, the way the correlation coefficient has been implemented in this thesis means that correlation between features is taken into account while the correlation between features and a class (e.g. cancer or normal) is not taken into account. Therefore, the EA is trying to learn information from the overall correlation of features in the dataset. As this correlation may not be significantly related to a particular class, the EA may be selecting features that are correlated for some other reason than the case under study. In a gene expression profile, many normal expression patterns will be present, as well expression patterns related to the case under study (e.g. cancer). Therefore,

it is possible that the EA is forced to search for expression patterns that are not correlated to the case under study. However, if this is the case, then this presents further evidence for the robustness of the EA as the EA is able to perform well although the mutation operator may be forcing the search towards unrelated features. This hypothesis can be investigated by using correlation coefficients calculated by taking the classification of the sample into account. Then, when the algorithm selects highly correlated features, there would be a certain degree of assurance that a particular feature would either be positively or negatively correlated to the case under study.

It is well known that DNA microarray data contains non-linear relationships between features [38]. Therefore, either a positive or negative bias towards features using a linear correlation measure may hinder the performance of the EA. Therefore, future research could concentrate on using a correlation measure that is capable of taking non-linear interactions into account (e.g. Spearman's r_s) in order to determine if correlation guided mutation approach could further improve the performance of the EA/Weighted Centroid algorithm.

Another factor that may lessen the effectiveness of the correlation guided mutation may be the length of the chromosome. With larger chromosomes, the methods implemented here have a more pronounced effect. With shorter chromosomes, they have a less pronounced effect. For example, when the length of the chromosome is two, the correlation guidance in removing a gene effectively becomes null. Therefore, another interesting approach would be to change the parameter α so that the algorithm keeps larger chromosomes for longer and as such gives the correlation guided mutation operator adequate time to guide the search process.

5.8 Conclusion

Chapter 4 of this thesis concluded that weighted k -NN is capable of outperforming classical k -NN. There was evidence in the literature to show that prior selection of features using correlation measures improves the performance of machine learning algorithms from a feature selection and classification point of view. Therefore, it was hypothesised that EA/ k -NN algorithm could perform better with the aid of correlation measures.

A literature search showed that there were three commonly used correlation co-

| Leukaemia dataset | | |
|-------------------------|---------------------|-------------|
| Generation | Validation accuracy | Length |
| 48 | 0.92 | 2 |
| 87 | 0.81 | 1 |
| 284 | 0.92 | 1 |
| 2051 | 0.88 | 1 |
| 173 | 0.65 | 1 |
| 219 | 0.88 | 1 |
| 472 | 0.69 | 1 |
| 1295 | 0.77 | 2 |
| Mean | 0.82 | 1.25 |
| SD | 0.10 | 0.46 |
| Prostate cancer dataset | | |
| Generation | Validation accuracy | Length |
| 2416 | 0.89 | 5 |
| 554 | 0.81 | 6 |
| 221 | 0.86 | 4 |
| 2791 | 0.83 | 3 |
| 8426 | 0.89 | 4 |
| 3008 | 0.86 | 5 |
| 6180 | 0.81 | 9 |
| 7 | 0.92 | 29 |
| Mean | 0.86 | 8.13 |
| SD | 0.04 | 8.63 |
| Breast cancer dataset | | |
| Generation | Validation accuracy | Length |
| 9658 | 0.52 | 4 |
| 3586 | 0.52 | 4 |
| 7942 | 0.64 | 6 |
| 752 | 0.61 | 4 |
| 2340 | 0.58 | 1 |
| 136 | 0.64 | 9 |
| 359 | 0.52 | 6 |
| 2579 | 0.61 | 2 |
| Mean | 0.58 | 4.50 |
| SD | 0.05 | 2.51 |
| Colon cancer dataset | | |
| Generation | Validation accuracy | Length |
| 8841 | 0.82 | 2 |
| 9876 | 0.77 | 4 |
| 5758 | 0.68 | 3 |
| 3430 | 0.55 | 2 |
| 9667 | 0.86 | 5 |
| 70 | 0.64 | 5 |
| 9914 | 0.59 | 1 |
| 3253 | 0.73 | 3 |
| Mean | 0.70 | 3.13 |
| SD | 0.11 | 1.46 |
| Ovarian cancer dataset | | |
| Generation | Validation accuracy | Length |
| 2298 | 0.95 | 2 |
| 993 | 0.95 | 2 |
| 7440 | 0.99 | 2 |
| 3149 | 0.99 | 5 |
| 1125 | 0.99 | 2 |
| 5273 | 0.98 | 4 |
| 2512 | 0.98 | 2 |
| 332 | 0.99 | 4 |
| Mean | 0.98 | 2.88 |
| SD | 0.02 | 1.25 |

Table 5.3: Overall results for correlation guided mutation with preference for strongly correlated feature subsets for the Evolutionary Algorithm/ k -Nearest Neighbours algorithm.

| Leukaemia dataset | | |
|-------------------------|---------------------|-------------|
| Generation | Validation accuracy | Length |
| 48 | 0.88 | 2 |
| 495 | 0.81 | 1 |
| 498 | 0.92 | 1 |
| 445 | 0.88 | 1 |
| 1505 | 0.65 | 1 |
| 3023 | 0.88 | 1 |
| 9769 | 0.73 | 2 |
| 398 | 0.77 | 2 |
| Mean | 0.82 | 1.38 |
| SD | 0.09 | 0.52 |
| Prostate cancer dataset | | |
| Generation | Validation accuracy | Length |
| 8369 | 0.64 | 3 |
| 48 | 0.83 | 2 |
| 4419 | 0.78 | 2 |
| 1681 | 0.86 | 4 |
| 878 | 0.92 | 7 |
| 4183 | 0.86 | 5 |
| 9977 | 0.75 | 3 |
| 5218 | 0.94 | 3 |
| Mean | 0.82 | 3.63 |
| SD | 0.10 | 1.69 |
| Breast cancer dataset | | |
| Generation | Validation accuracy | Length |
| 999 | 0.58 | 9 |
| 4488 | 0.48 | 6 |
| 7156 | 0.64 | 6 |
| 5889 | 0.61 | 6 |
| 9223 | 0.58 | 5 |
| 5084 | 0.42 | 5 |
| 5473 | 0.82 | 3 |
| 3675 | 0.52 | 1 |
| Mean | 0.58 | 5.13 |
| SD | 0.12 | 2.36 |
| Colon cancer dataset | | |
| Generation | Validation accuracy | Length |
| 2628 | 0.82 | 7 |
| 3147 | 0.77 | 3 |
| 9577 | 0.68 | 4 |
| 3630 | 0.73 | 4 |
| 2837 | 0.91 | 3 |
| 2408 | 0.68 | 3 |
| 347 | 0.77 | 1 |
| 887 | 0.91 | 2 |
| Mean | 0.78 | 3.38 |
| SD | 0.09 | 1.77 |
| Ovarian cancer dataset | | |
| Generation | Validation accuracy | Length |
| 2577 | 0.94 | 1 |
| 2396 | 0.99 | 2 |
| 333 | 0.98 | 2 |
| 4231 | 0.98 | 2 |
| 786 | 0.99 | 2 |
| 281 | 0.98 | 3 |
| 72 | 0.99 | 2 |
| 392 | 0.91 | 2 |
| Mean | 0.97 | 2.00 |
| SD | 0.03 | 0.53 |

Table 5.4: Overall results for correlation guided mutation approach with preference for loosely correlated feature subsets for the Evolutionary Algorithm/ k -Nearest Neighbours algorithm.

| Leukaemia dataset | | |
|-------------------------|---------------------|-------------|
| Generation | Validation accuracy | Length |
| 7897 | 0.88 | 2 |
| 507 | 0.81 | 2 |
| 1008 | 0.96 | 1 |
| 836 | 0.88 | 2 |
| 67 | 1.00 | 2 |
| 4818 | 0.81 | 1 |
| 88 | 0.92 | 7 |
| 991 | 0.85 | 1 |
| Mean | 0.89 | 2.25 |
| SD | 0.07 | 1.98 |
| Prostate cancer dataset | | |
| Generation | Validation accuracy | Length |
| 339 | 0.69 | 3 |
| 717 | 0.86 | 4 |
| 1757 | 0.78 | 1 |
| 155 | 0.89 | 4 |
| 7107 | 0.81 | 9 |
| 538 | 0.83 | 2 |
| 4215 | 0.81 | 6 |
| 2244 | 0.81 | 5 |
| Mean | 0.81 | 4.25 |
| SD | 0.06 | 2.49 |
| Breast cancer dataset | | |
| Generation | Validation accuracy | Length |
| 9785 | 0.70 | 5 |
| 1739 | 0.70 | 8 |
| 8914 | 0.64 | 5 |
| 8404 | 0.61 | 3 |
| 3470 | 0.52 | 1 |
| 7772 | 0.61 | 2 |
| 3831 | 0.76 | 2 |
| 316 | 0.52 | 5 |
| Mean | 0.63 | 3.88 |
| SD | 0.09 | 2.30 |
| Colon cancer dataset | | |
| Generation | Validation accuracy | Length |
| 1479 | 0.91 | 4 |
| 1637 | 0.82 | 2 |
| 2682 | 0.91 | 4 |
| 9810 | 0.55 | 2 |
| 5934 | 0.95 | 2 |
| 1594 | 0.73 | 2 |
| 34 | 0.77 | 1 |
| 1395 | 0.82 | 3 |
| Mean | 0.81 | 2.50 |
| SD | 0.13 | 1.07 |
| Ovarian cancer dataset | | |
| Generation | Validation accuracy | Length |
| 832 | 0.96 | 2 |
| 144 | 0.99 | 3 |
| 2195 | 0.96 | 3 |
| 6516 | 0.98 | 5 |
| 1229 | 0.99 | 2 |
| 4227 | 0.95 | 2 |
| 1366 | 1.00 | 2 |
| 30 | 0.99 | 5 |
| Mean | 0.98 | 3.00 |
| SD | 0.02 | 1.31 |

Table 5.5: Overall results for correlation guided mutation approach with preference for strongly correlated feature subsets for Evolutionary Algorithm/Weighted Centroid Classifier algorithm.

| | Leukaemia | Prostate | Breast | Colon | Ovarian |
|---|-------------|-------------|-------------|-------------|-------------|
| Baseline(EA/ k -NN) | 0.83 | 0.82 | 0.61 | 0.72 | 0.97 |
| EA/W- k -NN | 0.81 | 0.85 | 0.58 | 0.76 | 0.97 |
| EA/Weighted Centroid | 0.80 | 0.79 | 0.63 | 0.76 | 0.98 |
| EA/ k -NN & Correlation guided mutation with preference for the most correlated features | 0.82 | 0.86 | 0.58 | 0.70 | 0.98 |
| EA/ k -NN & Correlation guided mutation with preference for the least correlated features | 0.82 | 0.82 | 0.58 | 0.78 | 0.97 |
| EA/Weighted Centroid & Correlation guided mutation with preference for most correlated features | 0.89 | 0.81 | 0.63 | 0.81 | 0.98 |

Table 5.6: Comparison of correlation guided mutation approaches to other approaches used in this thesis. The best algorithm for each dataset is highlighted in boldface.

efficients. Out of these three, Person’s correlation coefficient was chosen for this study as it is potentially more powerful than the alternatives. A novel correlation guided mutation operator was then introduced. This mutation operator either added, removed or changed a gene in a chromosome using probabilities for each gene calculated with the help of Perason’s correlation coefficient.

Correlation coefficient was primarily used in two ways:

- Include the most strongly correlated features in the selected feature subset.
- Include a diverse set of loosely correlated features in the selected feature subset.

Both of these approaches were compared with the baseline algorithm (EA/ k -NN), the weighted algorithm (EA/W- k -NN) and the weighted centroid algorithm (EA/Weighted Centroid). The approach that had a preference for strongly correlated features in the selected feature subset performed slightly better than the approach that had a preference for selecting loosely correlated features in the selected feature subset.

The approach that had a preference for strongly correlated features was then implemented for the EA/Weighted Centroid algorithm. The EA/Weighted Centroid

algorithm, with the help of the correlation guided mutation operator, managed to outperform the other algorithms on four out of the five datasets tested in this thesis.

This Chapter adds more evidence in support of the conclusion made in Chapter 4 that stated that the weighted approaches were capable of outperforming non-weighted approaches. It also provides evidence to show that correlation measures can improve the performance of feature selection and classification algorithms. Areas for further research in the use of correlation measures in feature selection and classification have also been identified.

Chapter 6

Conclusions and Future Work

This thesis concentrated on research into FS and classification in predictive data mining, especially in large biological datasets such as DNA microarray data and proteomic analysis data using EAs. Particular attention was paid to the two-phase EA/ k -NN algorithm and an adaptive weights scheme for k -NN. Finally, an investigation was carried out into the possible use of correlation guided mutation in the EA/ k -NN algorithm.

6.1 Main Findings

6.1.1 Parameters and Multi-Objective Approach in Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm

Although results for the datasets used in this thesis have been reported elsewhere, it is difficult to do a direct comparison. This is due to the fact that different researchers use slightly different versions of the dataset and their validation methods also vary from one study to another. However, a comparison can nevertheless be made and results from this thesis are very competitive, if not better, compared than results published in other literature.

Properly tuned, the two-phase EA/ k -NN algorithm achieved very competitive results to those that were published in the literature for all three datasets tested in Chapter 3. The multi-objective version of the two-phase EA/ k -NN algorithm also achieved excellent results. The advantage of the multi-objective approach is

that there is no need to pre-tune some of the parameters (e.g. α that combines the classification error with the length of the chromosome).

These results have provided further evidence that the two-phase EA/ k -NN algorithm is clearly competitive with other methods when it comes to FS and classification.

However, the two-phase EA/ k -NN algorithm is sensitive to the way phase one is conducted. Phase one needs to avoid over-fitting while gathering informative genes so that phase two can build robust models. The termination point of phase one is therefore critical in the success of the complete algorithm. As a clear termination point could not be identified in a straightforward way, a new method for terminating the algorithm was introduced. This method tracks the classification accuracy of each gene that is used in the algorithm across all the generations of the first phase of the algorithm. This method avoided the requirement for tuning the termination point of the first phase of the algorithm.

6.1.2 An Adaptive Weights Scheme for k -Nearest Neighbours in Evolutionary Algorithm/ k -Nearest Neighbours Algorithm for Feature Selection

Although the two-phase EA/ k -NN algorithm looks promising for FS and classification in predictive data mining, further testing across five datasets showed that the single phase EA/ k -NN algorithm held a slight advantage. It may still be possible to set-up the two-phase EA/ k -NN algorithm in such a way that it clearly out-performs the EA/ k -NN algorithm across all five datasets. However, it was decided that single phase EA/ k -NN algorithm would be used for the remainder of the thesis due to the fact that it out-performed the two-phase algorithm.

The direct comparison between EA/ k -NN and EA/W- k -NN algorithms (using two weighted approaches) across the five datasets has shown that when properly tuned, the EA/W- k -NN algorithm is capable of out-performing the EA/ k -NN algorithm.

The direct application of adaptive weights to the k -NN classifier provided better or equal results on three out of the five datasets. However, a novel weighted centroid classifier managed to obtain better results for four out of the five datasets tested.

The weighted centroid classification technique introduced here has the advantage

that it is much less memory-intensive and several orders of magnitude faster than either the EA/ k -NN or the EA/W- k -NN algorithm. For example, on the ovarian cancer dataset, one run of the EA/W- k -NN algorithm took 44 hours to complete on the Beowulf cluster at Heriot-Watt University. The same run only took 2 hours to complete using the weighted centroid classification approach.

6.1.3 Correlation Guided Evolutionary Algorithm/ k -Nearest Neighbours for Feature Selection

There is evidence in the literature to suggest that prior FS leads to better performance in predictive data mining in large datasets [38, 27]. This is due to the fact that prior FS removes redundant and noisy features from high dimensional datasets. Statistical correlation is used widely in such efforts. However, there is very little evidence to suggest that correlation has been used in combination with EAs in FS and classification.

Therefore, a correlation guided EA was introduced in Chapter 5. This algorithm uses Pearson's product-moment correlation coefficient calculated on the entire training fold of the dataset to guide the mutation operator in the EA. The mutation operator added, removed or changed genes encoded in the chromosome with two approaches:

- Include the most strongly correlated features in the selected feature subset. With this approach, when removing a gene, loosely correlated genes in the chromosome had a higher chance of being removed. When adding a gene, strongly correlated genes in the dataset had a higher chance of being added to the chromosome. When changing a chromosome, loosely correlated genes in the chromosome had a higher chance of being replaced with strongly correlated features in the dataset.
- Include a diverse set of loosely correlated features in the selected feature subset. With this approach, genes in the chromosome that were strongly correlated had a higher chance of being removed. Loosely correlated genes in the dataset had a higher chance of being added to the chromosome. While changing a gene, the mutation operator ensured that strongly correlated genes encoded in the chromosome had a higher chance of being changed into a loosely correlated

feature in the dataset.

The EA/Weighted Centroid algorithm with correlation guided mutation performed better than the other algorithms on four out of the five datasets that were tested in this thesis. However, when the EA/Weighted Centroid algorithm was compared to the EA/Weighted Centroid algorithm with correlation guided mutation operator, the EA/Weighted Centroid algorithm with correlation guided mutation performed better on all five datasets. This provides further evidence in support of the argument that correlation measures can improve the performance of feature selection and classification algorithms.

6.2 Future Work

There are a number of areas presented in this thesis that would benefit from further work. This possible additional work is discussed in the following sections.

6.2.1 Setting up of the Two-Phase Evolutionary Algorithm/ k -Nearest Neighbours Algorithm

The two-phase EA/ k -NN algorithm has been shown to be clearly sensitive to some of the parameters and also the way the two phases are set up. Chapter 3 concluded that the two-phase algorithm is clearly better than the single phase algorithm. However, further testing in Chapter 4 revealed that single phase algorithm slightly outperformed the two-phase algorithm. As previous results have shown the two-phase algorithm to be promising, more research could be carried out to determine a better way of setting up the two phases so that the algorithm has a better chance of good performance across a range of datasets.

One of the ways of optimising the setting up of the two phases could be to try introducing different amounts of genes from phase one to the second phase. In this thesis, 10% of the genes from phase one was carried into phase two. As it is critical that informative genes are kept in the gene pool, further research can clarify whether taking a higher percentage of genes to the second phase would improve the performance of the two-phase EA/ k -NN algorithm.

A value of 1000 was used for the parameter α for all the datasets. This value was determined by testing three out of the five datasets. Further research could be

carried out to identify an appropriate value for α for each dataset. This is necessary as the value of α determines the length of the chromosome and how quickly the algorithm converges. A quickly converging algorithm in phase one is not desirable as the search would not be able to cover interesting areas of the landscape and therefore may lead to poor performance in phase two.

Another area of further research could be an investigation into the subsets of genes identified by different runs of the algorithm. The purpose of predictive data mining in biological datasets is to discover subsets of genes that have a high classification performance. If the algorithm finds similar subsets of genes across multiple runs, then it can be confidently concluded that these genes are in fact related to the case under study.

6.2.2 Multi-objective Approach

The multi-objective approach presented in Chapter 3 looked promising for feature selection and classification. However, it was decided that the single object approach would be taken forward in order to do a direct comparison with results presented by Juliusdottir et al. [38].

The advantage of the multi-objective approach is that the tuning of the parameter α is not needed. The multi-objective approach was only tested with termination strategy for phase one discussed in Chapter 3. This termination point also had to be tuned. However, there is no need for tuning this parameter in the strategy proposed in Chapter 4. Therefore, it would be interesting to investigate the performance of the multi-objective approach using the strategy proposed in Chapter 4.

6.2.3 Adaptive Weights

Adaptive weights, especially in the form of weighted centroid classifier introduced in Chapter 4 has been shown to out-perform the classical EA/ k -NN algorithm in feature selection and classification in five large biomedical datasets. As the adaptive weights strategy was adopted from (and modified in this thesis) the ALH algorithm introduced by Yang and Kecman [81], it would be interesting to do a direct comparison between the approach outlined in this thesis with that of Yang and Kecman [81]. This could be achieved by testing the same datasets that they tested and in the same way (e.g. validation method). This would enable the testing of the hypothesis that

a straightforward classifier (e.g. k -NN) is better than a sophisticated classifier (e.g. SVM) in the context of FS and classification.

One major problem with the adaptive weights approach is the time taken for calculating the weights. This was drastically reduced by optimising the algorithm and also by pre-calculating most of the weights. Although this speeded up the algorithm, a drawback is that if the dataset is very large (much larger than the ovarian dataset), it is possible that the algorithm may not have enough Random Access Memory (RAM) to work with. Therefore, further research could be carried out into a solution that is faster and less memory-intensive. One interesting approach that was tried briefly in order to cut down computation time was fitness inheritance [5]. Further research could be carried out on ways to apply fitness inheritance to the EA/W- k -NN algorithm.

As shown in section 4.7.2, the performance of EA/W- k -NN algorithm relies heavily on the value used for the parameter T . The optimal value for T varies depending on the dataset used. It is time-consuming and impractical to calculate T manually for each new dataset that is used with the algorithm. Therefore, a brief investigation was carried out into encoding T as part of the chromosome and letting it evolve.

For these experiments, a child chromosome inherited the T associated with the stronger of its two parents. If both parents were equal, then the child chromosome took the average T of its parent chromosomes. The T of the child chromosome was then mutated with a probability of 50% (mutate T 50% of the time).

The mutation operator generated a Gaussian random number and then scaled it using a pre-determined scaling factor. For example, if the scaling factor was set to 10, then the random numbers would be distributed between -10 and 10 with a higher probability of the random number being closer to 0. The random number was then shifted to be around a pre-determined mean. The modified Gaussian random number was then simply added to the T value of the child chromosome. With this method, the mean and the SD of the generated Gaussian random numbers could be tightly controlled. The investigation then concentrated on finding the best values for the mean and the scaling factor.

The results obtained from this investigation were not encouraging. There was no single set of values for the mean and the scaling factor that worked well across all the datasets. The aim of this investigation was to find a way to let T evolve so that no initial tuning of T would be required for a new dataset. However, if a series

of experiments had to be conducted to find an effective set of values for the mean and the scaling factor for T for each dataset, then this would mean that the research aim would not be met by this approach.

Another reason why this investigation did not meet the research aims could be due to the fact that the algorithm is very sensitive to the value of T . As shown in section 4.7.2, even a small deviation in the optimal value of T leads to a dramatic change in classification accuracies. The method for evolving T implemented in this thesis would continue to make changes to T and therefore would continue changing the fitness landscape from the point of view of the EA. This would make it difficult for the algorithm to converge on an optimal set of features.

As the brief investigation did not produce promising results, it was decided not to pursue it further due to other priorities and constraints. However, as the EA/ W - k -NN algorithm looks promising, it would be interesting to carry out further research into evolving T . This research could be aimed at achieving two objectives:

- Achieve better results from the algorithm with regards to classification accuracy and selected subset of features.
- Increase the amount of automation in the algorithm by evolving T so that the application of the algorithm for a new dataset is not as taxing as it is now.

6.2.4 Correlation Guided Mutation for Evolutionary Algorithm/ k -Nearest Neighbours Algorithm

In accordance with the evidence present in the literature, introduction of correlation measures improved the performance of the EA/Weighted Centroid algorithm. However, promising areas for further research have also been identified.

The most logical next step is to calculate the correlation coefficient taking into account the classification of the samples present in the training fold of the dataset.

Pearson's correlation coefficient was selected for this study. However, Pearson's correlation coefficient is incapable of taking non-linear interactions between features into account. It is well documented that DNA microarray data contains non-linear relationships. Therefore, a logical next step would be to test a correlation coefficient that can take non-linear relationships into account such as Spearman's rho coefficient.

Furthermore, as discussed in Chapter 5, the length of the chromosome has an effect on the effectiveness of the correlation guided mutation approach. Therefore, further research could be carried out into ways in which the chromosome could be kept artificially long for a certain number of generations.

Another interesting area of research would be to introduce the correlation guided approach to the cross-over operator.

6.3 Final Thoughts

This thesis has presented research into FS and classification in predictive data mining in large biomedical datasets using EAs. The two-phase EA/ k -NN algorithm introduced by Juliusdottir et al. [38] has been thoroughly investigated for parameter optimisation and the optimal way of setting up the two phases. An investigation into an adaptive weights scheme for the k -NN algorithm was carried out and a promising new weighted centroid classifier has been identified. Research was also presented into a new way of guiding the mutation operator (correlation guided) in an EA in FS and classification. Rich and promising areas of further research have also been identified which should add further value to techniques investigated and introduced in this thesis. Table 6.1 shows the best results for each algorithm for each dataset tested in this thesis.

| | Leukaemia | Prostate | Breast | Colon | Ovarian |
|--|-------------|-------------|-------------|-------------|-------------|
| Baseline(EA/k-NN) | 0.83 | 0.82 | 0.61 | 0.72 | 0.97 |
| EA/W-k-NN | 0.81 | 0.85 | 0.58 | 0.76 | 0.97 |
| EA/Weighted Centroid | 0.80 | 0.79 | 0.63 | 0.76 | 0.98 |
| EA/k-NN & Correlation guided mutation with preference for the most correlated features | 0.82 | 0.86 | 0.58 | 0.70 | 0.98 |
| EA/k-NN & Correlation guided mutation with preference for the least correlated features | 0.82 | 0.82 | 0.58 | 0.78 | 0.97 |
| EA/Weighted Centroid & Correlation guided mutation with preference for most correlated features | 0.89 | 0.81 | 0.63 | 0.81 | 0.98 |

Table 6.1: The best results from all the algorithms tested in this thesis. The best algorithm for each dataset is highlighted in boldface.

Bibliography

- [1] C. Abdullah Konaka, D. W. Coitb, and A. E. Smithc. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering and System Safety*, 91:992–1007, 2006.
- [2] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences of the United States of America*, 96(12):6745–6750, 1999.
- [3] R. P. Anderson, D. Lew, and A. T. Peterson. Evaluating predictive models of species distributions: criteria for selecting optimal models. *Ecological Modelling*, 162(3):211–232, 2003.
- [4] Y. Bangpeng and L. Shao. ANMM4CBR: a case-based reasoning method for gene expression data classification. *Algorithms for Molecular Biology*, 5, 2010.
- [5] R. Barbour, D. Corne, and J. McCall. Accelerated optimisation of chemotherapy dose schedules using fitness inheritance. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8. IEEE, 2010.
- [6] V. Bolón-Canedo, N. Sánchez-Marño, A. Alonso-Betanzos, J. Benítez, and F. Herrera. A review of microarray datasets and applied feature selection methods. *Information Sciences*, 282:111–135, 2014.
- [7] J. Branke. Creating robust solutions by means of evolutionary algorithms. In *Parallel Problem Solving from NaturePPSN V*, pages 119–128. Springer, 1998.
- [8] J. Branke and C. Schmidt. Faster convergence by means of fitness estimation. *Soft Computing*, 9(1):13–20, 2005.

- [9] P. Burman. A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. *Biometrika*, 76(3):503–514, 1989.
- [10] A. Butte. The use and analysis of microarray data. *Nature Reviews Drug Discovery*, 1(12):951–960, 2002.
- [11] D. W. Corne, J. D. Knowles, and M. J. Oates. The pareto envelope-based selection algorithm for multiobjective optimization. In *Parallel Problem Solving from Nature PPSN VI*, pages 839–848. Springer, 2000.
- [12] K. P. Dahal and J. McDonald. Generator maintenance scheduling of electric power systems using genetic algorithms with integer representation. 1997.
- [13] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1(3):131–156, 1997.
- [14] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. *Lecture notes in computer science*, 1917:849–858, 2000.
- [15] R. Debnath and T. Kurita. An evolutionary approach for gene selection and classification of microarray data based on SVM error-bound theories. *Biosystems*, 2010.
- [16] J. Dougherty, R. Kohavi, M. Sahami, et al. Supervised and unsupervised discretization of continuous features. In *ICML*, pages 194–202, 1995.
- [17] E. Falkenauer. A new representation and operators for genetic algorithms applied to grouping problems. *Evolutionary computation*, 2(2):123–144, 1994.
- [18] E. Fix and J. L. Hodges. Discriminatory analysis - nonparametric discrimination - consistency properties. *International Statistical Review*, 57(3):238–247, 1989.
- [19] C. M. Fonseca and P. J. Fleming. Multiobjective genetic algorithms. In *Genetic Algorithms for Control Systems Engineering, IEE Colloquium on*, pages 6–1. IET, 1993.

- [20] T. D. Gautheir. Detecting trends using spearman's rank correlation coefficient. *Environmental Forensics*, 2(4):359–362, 2001.
- [21] D. E. Goldberg and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. *Foundations of genetic algorithms*, 1:69–93, 1991.
- [22] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.
- [23] J. Gosling. *The Java language specification*. Addison-Wesley Professional, 2000.
- [24] J. J. Grefenstette and J. M. Fitzpatrick. Genetic search with approximate function evaluations. In *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, pages 112–120, 1985.
- [25] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422, 2002.
- [26] P. Hajela and C.-Y. Lin. Genetic search strategies in multicriterion optimal design. *Structural optimization*, 4(2):99–107, 1992.
- [27] M. A. Hall and L. A. Smith. Feature subset selection: a correlation based filter approach. 1997.
- [28] J. Hauke and T. Kossowski. Comparison of values of pearson's and spearman's correlation coefficients on the same sets of data. *Quaestiones geographicae*, 30(2):87–93, 2011.
- [29] M. Hilbert and P. López. The worlds technological capacity to store, communicate, and compute information. *Science*, 332(6025):60–65, 2011.
- [30] J. Horn and D. E. Goldberg. Genetic algorithm difficulty and the modality of fitness landscapes. *Foundations of genetic algorithms*, 3:243–269, 1995.
- [31] J. Horn, N. Nafpliotis, and D. E. Goldberg. A niched pareto genetic algorithm for multiobjective optimization. In *Evolutionary Computation, 1994. IEEE*

- World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, pages 82–87. Ieee, 1994.
- [32] C. Z. Janikow and Z. Michalewicz. An experimental comparison of binary and floating point representations in genetic algorithms. In *ICGA*, pages 31–36, 1991.
- [33] T. Jirapech-Umpai and S. Aitken. Feature selection and classification for microarray data analysis: evolutionary methods for identifying predictive genes. *BMC Bioinformatics*, 6:11, 2005.
- [34] D. Joanes and C. Gill. Comparing measures of sample skewness and kurtosis. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 47(1):183–189, 1998.
- [35] J. B. Johnson and K. S. Omland. Model selection in ecology and evolution. *Trends in Ecology & Evolution*, 19(2):101–108, 2004.
- [36] N. L. Johnson, S. Kotz, and N. Balakrishnan. *Continuous univariate distributions. Vol. 1*, volume 2. Wiley New York, 1995.
- [37] M. Johnston. Gene chips: array of hope for understanding gene regulation. *Current Biology*, 8(5):R171–R174, 1998.
- [38] T. Juliusdottir, E. Keedwell, D. Corne, and A. Narayanan. Two-phase ea/k-nn for feature selection and classification in cancer microarray datasets. In *Computational Intelligence in Bioinformatics and Computational Biology, 2005. CIBCB '05. Proceedings of the 2005 IEEE Symposium on*, pages 1–8, 2005.
- [39] M. G. Kendall. Rank correlation methods. 1948.
- [40] J. Knowles and D. Corne. The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 1. IEEE, 1999.
- [41] R. Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, volume 14, pages 1137–1145, 1995.

- [42] R. Kohavi and D. Sommerfield. Feature subset selection using the wrapper method: overfitting and dynamic search space topology. In *KDD*, pages 192–197, 1995.
- [43] P. Langley and S. Sage. Induction of selective bayesian classifiers. In *Proceedings of the Tenth International Conference on Uncertainty in Artificial Intelligence*, pages 399–406. Morgan Kaufmann Publishers Inc., 1994.
- [44] M. Laumanns, E. Zitzler, and L. Thiele. On the effects of archiving, elitism, and density based selection in evolutionary multi-objective optimization. In *Evolutionary multi-criterion optimization*, pages 181–196. Springer, 2001.
- [45] J. Lee Rodgers and W. A. Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.
- [46] L. P. Li, T. A. Darden, C. R. Weinberg, A. J. Levine, and L. G. Pedersen. Gene assessment and sample classification for gene expression data using a genetic algorithm/k-nearest neighbor method. *Combinatorial Chemistry & High Throughput Screening*, 4(8):727–739, 2001.
- [47] L. P. Li, D. M. Umbach, P. Terry, and J. A. Taylor. Application of the ga/knn method to seldi proteomics data. *Bioinformatics*, 20(10):1638–1640, 2004.
- [48] L. P. Li, C. R. Weinberg, T. A. Darden, and L. G. Pedersen. Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the ga/knn method. *Bioinformatics*, 17(12):1131–1142, 2001.
- [49] B. Liu, Q. H. Cui, T. Z. Jiang, and S. D. Ma. A combinational feature selection and ensemble neural network method for classification of gene expression data. *BMC Bioinformatics*, 5:12, 2004.
- [50] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):491–502, 2005.
- [51] H. Lu and G. G. Yen. Rank-density-based multiobjective genetic algorithm and benchmark test function study. *Evolutionary Computation, IEEE Transactions on*, 7(4):325–343, 2003.

- [52] Z. Mei, Q. Shen, and B. X. Ye. Hybridized knn and svm for gene expression data classification. *Life Science Journal-Acta Zhengzhou University Overseas Edition*, 6(1):61–66, 2009.
- [53] J. Moors. The meaning of kurtosis: darlington reexamined. *The American Statistician*, 40(4):283–284, 1986.
- [54] P. Mundra and J. Rajapakse. SVM-RFE with relevancy and redundancy criteria for gene selection. *Pattern Recognition in Bioinformatics*, pages 242–252, 2009.
- [55] T. Murata and H. Ishibuchi. Moga: Multi-objective genetic algorithms. In *Evolutionary Computation, 1995., IEEE International Conference on*, volume 1, page 289. IEEE, 1995.
- [56] L. Nanni. A novel ensemble of classifiers for protein fold recognition. *Neurocomputing*, 69(16-18):2434–2437, 2006.
- [57] L. Nanni and A. Lumini. An ensemble of k-local hyperplanes for predicting protein-protein interactions. *Bioinformatics*, 22(10):1207, 2006.
- [58] T. K. Paul and H. Iba. Extraction of informative genes from microarray data. In H. G. Beyer, editor, *Genetic and Evolutionary Computation Conference*, pages 453–460, Washington, DC, 2005. Assoc Computing Machinery.
- [59] A. C. Pease, D. Solas, E. J. Sullivan, M. T. Cronin, C. P. Holmes, and S. Fodor. Light-generated oligonucleotide arrays for rapid dna sequence analysis. *Proceedings of the National Academy of Sciences*, 91(11):5022–5026, 1994.
- [60] S. Peng, Q. Xu, X. B. Ling, X. Peng, W. Du, and L. Chen. Molecular classification of cancer types from microarray data using the combination of genetic algorithms and support vector machines. *FEBS letters*, 555(2):358–362, 2003.
- [61] E. F. Petricoin, A. M. Ardekani, and B. A. Hitt. Use of proteomic patterns in serum to identify ovarian cancer (vol 359, pg 572, 2002). *Lancet*, 364(9434):582–582, 2004.
- [62] W. F. Punch, E. D. Goodman, P. Min, C. S. Lai, P. Hovland, and R. Enbody. Further research on feature selection and classification using genetic algorithms. *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 557–564, 1993.

- [63] J. M. Raser and E. K. O'Shea. Noise in gene expression: origins, consequences, and control. *Science*, 309(5743):2010–2013, 2005.
- [64] M. L. Raymer, W. E. Punch, E. D. Goodman, L. A. Kuhn, and A. K. Jain. Dimensionality reduction using genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 4(2):164–171, 2000.
- [65] N. M. Razali and Y. B. Wah. Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of Statistical Modeling and Analytics*, 2(1):21–33, 2011.
- [66] M. Rogati and Y. Yang. High-performing feature selection for text classification. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, pages 659–661. ACM, 2002.
- [67] D. Rowntree. *Statistics without tears*. Penguin, 2000.
- [68] Y. Saeys, I. Inza, and P. Larranaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.
- [69] M. Sardana, R. Agrawal, and B. Kaur. An incremental feature selection approach based on scatter matrices for classification of cancer microarray data. *International Journal of Computer Mathematics*, (ahead-of-print):1–19, 2014.
- [70] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms, Pittsburgh, PA, USA, July 1985*, pages 93–100, 1985.
- [71] W. Siedlecki and J. Sklansky. A note on genetic algorithms for large scale feature selection. *Pattern Recognition Letters*, 10(5):335–347, 1989.
- [72] D. Singh, P. G. Febbo, K. Ross, D. G. Jackson, J. Manola, C. Ladd, P. Tamayo, A. A. Renshaw, A. V. D'Amico, J. P. Richie, E. S. Lander, M. Loda, P. W. Kantoff, T. R. Golub, and W. R. Sellers. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, 1(2):203–209, 2002.
- [73] R. E. Smith, B. A. Dike, and S. Stegmann. Fitness inheritance in genetic algorithms. In *Proceedings of the 1995 ACM symposium on Applied computing*, pages 345–350. ACM, 1995.

- [74] C. Spearman. The proof and measurement of association between two things. *The American journal of psychology*, 15(1):72–101, 1904.
- [75] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*, 2(3):221–248, 1994.
- [76] D. A. Van Veldhuizen and G. B. Lamont. Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evolutionary computation*, 8(2):125–147, 2000.
- [77] L. J. van’t Veer, H. Dai, M. J. Van De Vijver, Y. D. He, A. A. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveen, et al. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–536, 2002.
- [78] P. Vincent and Y. Bengio. K-local hyperplane and convex distance nearest neighbor algorithms. In *NIPS*, pages 985–992, 2001.
- [79] S. M. Weiss. *Predictive data mining: a practical guide*. Morgan Kaufmann, 1998.
- [80] S. Xiaoqiao and L. Yaping. Gene expression data classification using svm-knn classifier. In *Intelligent Multimedia, Video and Speech Processing, 2004. Proceedings of 2004 International Symposium on*, pages 149–152, 2004.
- [81] T. Yang and V. Kecman. Adaptive local hyperplane classification. *Neurocomputing*, 71(13-15):3001–3004, 2008.
- [82] G. G. Yen and H. Lu. Dynamic multiobjective evolutionary algorithm: adaptive cell-based rank and density estimation. *Evolutionary Computation, IEEE Transactions on*, 7(3):253–274, 2003.
- [83] L. Yu and H. Liu. Feature selection for high-dimensional data: a fast correlation-based filter solution. In *ICML*, volume 3, pages 856–863, 2003.
- [84] J. H. Zar. Significance testing of the spearman rank correlation coefficient. *Journal of the American Statistical Association*, 67(339):578–580, 1972.
- [85] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang. Multi-objective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1(1):32–49, 2011.

- [86] Z. Zhu, S. Jia, and Z. Ji. Towards a memetic feature selection paradigm. *IEEE Computational Intelligence Magazine*, 5(2):41–53, 2010.
- [87] E. Zitzler, M. Laumanns, L. Thiele, E. Zitzler, E. Zitzler, L. Thiele, and L. Thiele. Spea2: Improving the strength pareto evolutionary algorithm, 2001.
- [88] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *evolutionary computation, IEEE transactions on*, 3(4):257–271, 1999.